As per Revised Syllabus of BTE

DIPLOMA COURSE IN KARNATAKA

Semester - V (CS&E)

WEB PROGRAMMING

Mrs. Anuradha A. Puntambekar

M.E. (Computer)
Formerly Assistant Professor in
P.E.S. Modern College of Engineering, Pune

TECHNICAL PUBLICATIONS
An Up-Thrust for Knowledge

Price : ₹ 225/-

ISBN 978-93-332-1565-7

WEB PROGRAMMING

Semester - V (CS&E)

First Edition : June 2017

Published by :

TECHNICAL PUBLICATIONS™ Amit Residency, Office No.1, 412, Shaniwar Peth, Pune - 411030, M.S. INDIA.
Ph.: +91-020-24495496/97, Telefax : +91-020-24495497
Email : technical@vtobooks.com Website : www.technicalpublications.org

Printer :

Rachana Offset
S.no.15, Sukhsagar Nagar,
Waghjai Chowk, Arihant Marg, Katraj,
Pune - 411 046

Price : ₹ 225/-

ISBN 978-93-332-1565-7

9 789333 215657

DTEK 15

PREFACE

The importance of **Web Programming** is well known in various engineering fields. Overwhelming response to my books on various subjects inspired me to write this book. The book is structured to cover the key aspects of the subject **Web Programming**.

The book uses plain, lucid language to explain fundamentals of this subject. The book provides logical method of explaining various complicated concepts and stepwise methods to explain the important topics. Each chapter is well supported with necessary illustrations, practical examples and solved problems. All the chapters in the book are arranged in a proper sequence that permits each topic to build upon earlier studies. All care has been taken to make students comfortable in understanding the basic concepts of the subject.

Representative questions have been added at the end of each section to help the students in picking important points from that section.

The book not only covers the entire scope of the subject but explains the philosophy of the subject. This makes the understanding of this subject more clear and makes it more interesting. The book will be very useful not only to the students but also to the subject teachers. The students have to omit nothing and possibly have to cover nothing more.

I wish to express my profound thanks to all those who helped in making this book a reality. Much needed moral support and encouragement is provided on numerous occasions by my whole family. I wish to thank the **Publisher** and the entire team of **Technical Publications** who have taken immense pain to get this book in time with quality printing.

Any suggestion for the improvement of the book will be acknowledged and well appreciated.

Author
A. A. Puntambekar

Dedicated to God.

SYLLABUS

Web Programming (15CS52T)

UNIT I : Fundamentals and Introduction to XHTML (Chapter-1)

Fundamentals - A brief introduction to Internet, Origins, What the Internet Is, Internet Protocol Addresses, Domain Names, The World Wide Web, Origins, Web or Internet, Web browsers, Web servers, Web Server Operations, General Server Characteristics, Apache, IIS, Uniform Resource Locators, URL Formats, URL Paths, Multipurpose Internet Mail Extensions, Type Specification, Experimental Documental Types, Hypertext Transfer Protocol, The Request Phase, The Response Phase, Security, The web Programmers Toolbox

Introduction to XHTML - Syntactic differences between HTML and XHTML

UNIT II : JavaScript and XHTML Documents and Dynamic Documents with JavaScript (Chapters-2, 3)

JavaScript and XHTML documents - The JavaScript Execution Environment, The Document Object Model, Element Access in JavaScript, Events & Event Handling, Basic Concepts of Event handling, Events, Attributes & Tags, Handling Events from Body Elements, Handling Events from Button Elements, Handling Events from Textbox & password Elements, The Focus Event, Validating from Input, The DOM 2 Event Model, Event Propagation, Event handler registration, An Example of the DOM 2 Event Model, The Navigator Object, DOM Tree Traversal and Modification, DOM Tree Traversal, DOM Tree Modification.

Dynamic documents with JavaScript - Introduction, Positioning Elements, Absolute Positioning, Relative Positioning, Static Positioning, Moving Elements, Element Visibility, Changing Colors & Fonts, Changing Colors, Changing Fonts, Dynamic Contents, Stacking Elements, Locating the Mouse Cursor, Reacting to the Mouse Click, Slow Movement of Elements, Dragging & Dropping Elements.

UNIT III : Introduction to XML (Chapter-4)

Introduction, The Syntax of XML, XML Document Structure, Document Type Definitions, Declaring Elements, Declaring Attributes, Declaring Entities, A Sample DTD, Internal & External DTDs, Namespaces, XML Schema, Schemas Fundamentals, Defining the Schema,

Defining the Schema Instances, An Overview of Data types, Simple Types, Complex Types, Displaying Raw XML Documents, Displaying XML Documents with CSS.

UNIT IV : Introduction to PHP (Chapters-5)

Introduction to PHP - Origins and Uses of PHP, Overview of PHP, General Syntactic Characteristics, Primitives, Operations and Expressions, Variables, Integer Type, Double Type, String Type, Boolean Type, Arithmetic Operations & Expressions, String Operations, Scalar Type conversions, Output, Control statements, Relational Operators, Boolean Operators, Selection Statements, Loop statements, An Example, Arrays, Array Creation, Accessing array Elements, Functions for Dealing with Arrays, Sequential Access to Array Elements, Sorting Arrays, Functions, General Characteristics of Functions, Parameters, The scope of Variables, The Lifetime of Variables, Pattern Matching, Form Handling, Files, Opening and Closing Files, Reading from a File, Writing to a File, Locking Files, Cookies, Introduction to Cookies, PHP Support for Cookies, Session Tracking.

UNIT V : Database Access through the Web (Chapters-6)

Database Access with PHP & MySQL, Potential Problems with Special Characters, Connecting to MySQL & Selecting the Database, Requesting MySQL Operations, A PHP/ MySQL Examples, Database Access with JDBC & MySQL, JDBC & MySQL, Metadata, Examples.

UNIT VI : Java Web Software (Chapter-7)

Introduction to Servlets, Overview, Details, Servlet Containers, The NetBeans IDE, Storing information on Clients, Cookies, Servlet support for Cookies, Examples, JavaServer Pages(JSP), Motivation for JSP, JSP Documents, The Expression Language, The JSTL control action elements, JavaBeans, Model-View-Controller Application Architecture, JavaServer Faces, The tag libraries, JSF event handling, An example application.

TABLE OF CONTENTS

1 Fundamentals and Introduction to XHTML

Syllabus

Fundamentals *- A brief introduction to Internet, Origins, What the Internet Is, Internet Protocol Addresses, Domain Names, The World Wide Web, Origins, Web or Internet, Web browsers, Web servers, Web Server Operations, General Server Characteristics, Apache, IIS, Uniform Resource Locators, URL Formats, URL Paths, Multipurpose Internet Mail Extensions, Type Specification, Experimental Documental Types, Hypertext Transfer Protocol, The Request Phase, The Response Phase, Security, The web Programmers Toolbox*

Introduction to XHTML *- Syntactic differences between HTML and XHTML*

Contents

1.1 A Brief Introduction to Internet

Mar. 12, Marks 5

This is a chapter in which we will discuss all the Internet related concepts. First of all we will understand "what is internet and what is www". Then we will discuss the notion of Web servers and Web browsers with the help of examples. URL, is the most commonly used terminology in the world of WWW, we will discuss this term as well. Then we will be learning MIME and HTTP.

1.1.1 Origin

- The U.S. Department of Defense **(DoD)** decided to build a large scale computer network in 1960.
- This type of networking was developed for establishing the communication among the computers, accessing of remote computers and for sharing each other's program.
- The most important requirement of this type of computer networking was **robustness**. Robustness in networking means even if one of the computer in the network gets failed the network should continue its work.
- The first node of this network was established in 1969.
- The DoD's Advanced Research Project Agency **(ARPA)** funded this project hence this network was renamed as **ARPANet.**
- Although this network was established for communication, it was connected only to the laboratories and to the ARPA funded projects. The universities and other important organizations were not connected with the ARPANet. Hence many small networks were developed for their own needs.
- In 1986, a national network **NSFnet** was created. The NSFnet connected five universities which were funded by the NSF itself. Soon afterwards it was available to other universities and research laboratories. The network continued to grow very rapidly. By 1992, NSFnet connected around 1 million computers around the world.
- Thus NSFnet and ARPAnet collectively was known as **Internet.**

1.1.2 What the Internet is ?

Definition : Internet is global system in which millions of computers are connected together. It is basically a network of networks.

- Using internet many people can share resources and can communicate with each other. To have internet service one must go to the service providers. That means your computer must be connected to the Internet Service Providers (ISP) through cables modem, phone-line modem or DSL.

- There are some privately owned internet service providers from which we can hire the internet services.

1.1.3 Internet Protocol (IP) Address

Definition : Each host on a network is assigned a unique **32-bit logical address** that is divided into two main parts : the **network number** and the **host number**. This address is called IP address.

- The IP address is grouped four into 8 bits separated by dots. Each bit in the octet has binary weight

2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0

↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑

128 64 32 16 8 4 2 1

- As we know the IP address is divided in two categories network number and host number.

Network number	Host number

There are **5 classes** based on two categories viz. A, B, C, D and E.

IP Address Class	Format	Range	Purpose
Class A	N.H.H.H	1 to 126	Very few large organisations use this class addressing.
Class B	N.N.H.H	128 to 191	Medium size organisations use this addressing.
Class C	N.N.N.H	192 to 223	Relatively small organisations use this class.
Class D	-	224 to 239	This class address is used for multicast groups.
Class E	-	240 to 254	This class addressing is reserved for experimental purpose.

- Here N stands for network number and H stands for host number. For instance in class C first three octets are reserved for network address and last 8 bits denote host address.

1.1.4 Domain Names

- Each computer is assigned by a unique numerical address called IP address. This is because computer can remember the numerical addresses but humans can not remember numerical addresses, but they can remember names in better way.
- Hence the concept of **domain name** has come up.

Definition : Domain names are the unique names of host computers that can be used instead of numerical addresses.

- For example instead of 192.168.1.92 we can use www.mycomp.com
- In Domain Name Servers, instead of using the IP address the name of the computer is used to access that computer system. But two names can be the same. Hence to uniquely identify your computer the name must be created using **DNS hierarchy**.
- Some commonly used domain names are

Domain names	Purpose
com	Commercial organization
gov	Government organizations
edu	Educational institutes/organizations
int	International organization
net	Network group
org	Non profit organization
mil	Military group / organizations
in	Sub domain name used to refer India
uk	Sub domain name used to refer United Kingdom
jp	Sub domain name used to refer Japan

- The Internet logically arranges the domain names in an hierarchical form.
- There are some top level DNS such as com, org, edu, mil, net, uk, in and so on. Then each domain name is further divided into sub-domains then sub-sub-domains and so on.
- For example the complete path for **http://www.cse.tec.ac.in** can be uniquely traced out with the help of domain name space.

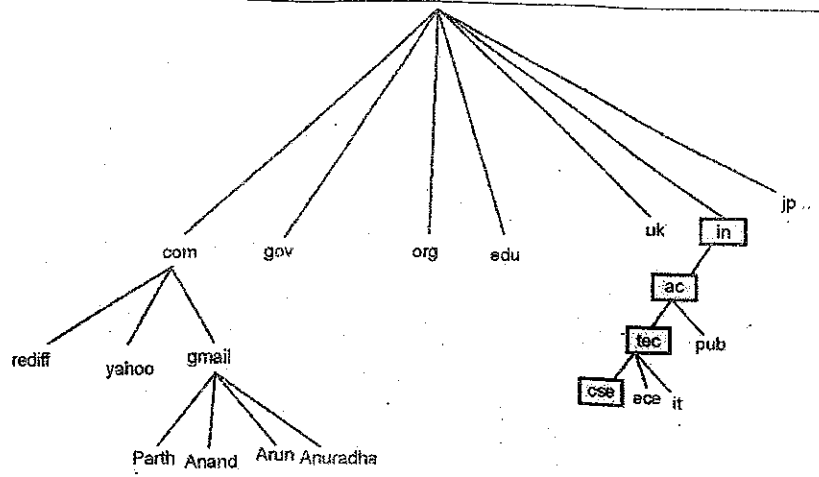


Fig. 1.1.1 Domain name space

Review Questions

1. *Explain domain names with an example.* Question Bank - Marks 5
2. *Define Internet.* Question Bank, Model Question Paper, Mark 1
3. *Define Internet. Explain briefly domain name.* May 12, Marks 5

1.2 WWW

- World Wide Web (WWW) is collections of software and corresponding protocols used to access the resources over the network.
- The world wide web is a **information system** in which various documents containing information are interlinked together. User can access this information or write the information via computers. This information is typically stored on the web pages and through web browsers we can access these web pages.
- The web pages may contain the information in the form of text, audio, video, images and graphics. We can navigate between the web pages using hyperlinks.

1.2.1 Origins

- The concept of WWW was introduced by **Sir Tim Berners-Lee** the contractor at **the European Organization for Nuclear Research (CERN)**, Switzerland in 1980. He built a personal database of people and software models and used hypertext so that each new page of information was linked to an existing page.
- In 1990, Berners-Lee introduced the tools such as HyperText Transfer Protocol (HTTP), HyperText Markup Language (HTML) and the web browser. This is the time when the first website **http://info.cern.ch/** was built.

- During 1992-1995, along with HTTP protocol a new protocol named **Gopher protocol** which provided access to content through hypertext menus presented as a file system rather than through HTML files. In 1993 a new web browser with graphical user interface **Mosaic** got introduced.
- In 1994, the **World Wide Web Consortium** (W3C) was founded by Berners-Lee at the **Massachusetts Institute of Technology** (MIT) with support from the **Defense Advanced Research Projects Agency (DARPA)**. This organisation was built for creating standards and recommendations to improve the quality of the Web. **Berners-Lee** made the Web available freely, with no patents. The World Wide Web Consortium (W3C) decided that their standards must be based on royalty-free technology, so they can be easily adopted by anyone. And then at the end of 1994 a large number of websites got activated with popular web services.
- During 1996-1998, trade marketing started using WWW. The term E-commerce got introduced during this period only.
- During 1999-2000, many entrepreneurs started selling their ideas using the **dotcom** boom.
- From 2002- till date, the WWW has got an evolving nature due to various development such as online-booking, efficient search engines and agent based technologies, Facebook, social networking sites and so on.

1.2.2 Web or Internet ?

- The term internet and WWW is often used interchangeably, but these are two different terms.
- The **internet** is collection of computers and other devices (such as printers, scanners etc.) connected together whereas World Wide Web (WWW) is collections of software and corresponding protocols used to access the resources over the network.
- The world wide web contains huge amount of documents, images and other resources which can be accessed using the hyperlinks.
- Thus people use internet through the Web.

1.3 Web Browser

May-14, Marks 7

Definition : Web browser is a application software which people use for retrieving, presenting and traversing information present on the web.

- Various commonly used web browsers are -
  1. Internet explorer 2. Mozilla Firefox

3. Google Chrome
4. Opera
5. Netscape Navigator

Review Question

1. *Define Web Browser* Question Bank, Model Question Paper, May-14, Mark 1

1.4 Web Servers

May-13, 14, Dec.-12, 16, Marks 5

Definition : Web server is a program that processes requests made by clients using HTTP protocol. It responds the clients by presenting requested web pages or files.

1.4.1 Web Server Operations

- The web server and web 'client communicate with each other using HTTP protocol.
- When web server starts its execution, it **informs operating system** that it is now ready to accept the requests on specific port.
- Web client opens a network connection with web server and sends the request to web server. The web server **listens** the web client requests on particular port.
- The **primary task** of web server is to monitor communications on specific port, accept HTTP command through that port and perform operations specified by these commands.
- When the request from client is received by the server it is translated either in **file name** or a **program name**. If it is a **filename**, then the requested file is transferred to the client. And if it is a **program name**, then the program is executed and its output is given to the client.

1.4.2 General Server Characteristics

File structure :

- There are two types of directories for the server. The roots of these directories are **document root** and **server root**. These directories and the subdirectories store the software required for server execution.
- The files that are stored directly in the document root directory are available to the clients using the top-level URL. Normally the clients do not access the document root directly.
- Normally server stores the documents that are readable to its client outside the document root.

- The **virtual document trees** are the areas from which the server can serve the documents to its clients.
- If the documents are stored in the subdirectories then client can refer to these web documents using the URL with a particular file path to that directory from the document root directory.
- Some servers allow the access to the web documents that are in the document root of other machine. Such servers are called **proxy servers.**
- Web servers support **various protocols** such as HTTP, FTP, Gopher, News and mail.
- All the web servers can interact with the database systems with the help of Common Gateway Interface (CGI) or server side script.

1.4.3 Apache

- It is an excellent server because of its two important features : **Reliability** and **Efficiency**.
- Secondly it is more popular because it is an **open source software**. That means it is freely available to anybody. Apache web server is best suitable for UNIX systems but it can also be used on Windows box. The apache web server can be configured as per the requirements using the file **httpd.conf.** This file is present in the Apache software package.

1.4.4 IIS

- The **Internet Information Services** or **Internet Information Server** is a kind of web server provided by Microsoft. This server is most popular on Windows platform.
- Following are some differences between Apache and IIS servers

Sr. No.	Apache Web Server	IIS Web Server
1.	Apache web server is useful on both Unix based systems and on Windows platform.	IIS web server is used on Windows platform.
2.	It is an open source product.	It is a vendor specific product and can be used on windows products only.
3.	The Apache web server can be controlled by editing the configuration file *httpd.conf*.	For IIS server, the behaviour is controlled by modifying the window based management programs called IIS snap in. We can access IIS snap in through the Control Panel - Administrative Tools.

Review Questions

1. *Define web server.* Question Bank, May 14, Marks 1
2. *Explain the operation of web server.* Question Bank, May 13, Dec. 16, Marks 5
3. *Illustrate the general server characteristics.* Question Bank, Dec. 12, 16, Marks 5
4. *Explain the file structure of web server.* Question Bank, Marks 5
5. *Explain the operation of web server with its file structure.* Question Bank, Marks 10

1.5 Uniform Resource Locator

Dec. 13, May 16, Marks 5

For identifying the documents on the internet the uniform or universal resource Locator i.e. URL is used.

There is variety of URL depending upon the type of resources.

1.5.1 URL Formats

- The general format of URL is –

Scheme:Address

That is

protocol://username@hostname/path/filename

- The scheme specifies the **communication protocol**. Different schemes have different schemes have different forms of addresses.
- Various schemes that are used are **http, ftp**, gopher, file, mailto, news and so on.
- The most commonly used protocol for web browser and web server communication is **hypertext transfer protocol(HTTP).** This protocol is based on request-response mechanism. This protocol handles the documents that are created using eXtensible Hypertext Markup Language (XHTML). Using **http** the **Address** part of URL can be written as follows –

//Domain name/path_to_Document

- **file** is another most commonly used scheme in URL. This protocol allows to reside the document in the client's machine from which the web browser is making out the demand. Due to this the actual document is not available only for its visibility but it can be tested. Using **file** the Address part of URL can be written as follows –

file://path-to-document

- The **hostname** is the name of the server computer that stores the web documents.

- The default port number for the http protocol is 80.
- Any URL does not allow the spaces in it. But there are some special characters that can be present in the URL. These characters are – ampersand & or percentage % .

1.5.2 URL Paths

- The path to the web document is similar to the path to the particular file present in the folder. In this path the directory names and files are separated by the separator characters. The character that is used as a separator is **slash.** Unix system uses forward slash whereas the windows system makes use of backward slash. For example -

```
http://www.mywebsite.com/mydocs/index.html
```

- The URL path that includes all the directories along the path to the file is called the **complete path.**
- Sometimes the base URL path is specified in the configuration file of the server. In such a case we need not have to specify the complete path for accessing the particular file such a path is called the **partial path.** For example -

```
http://www.mywebsite.com
```

This indicates that the file mydocs/index.html is specified in the configuration file.

Review Questions

1. *Discuss URL format with its different paths.* Question Bank, Marks 5
2. *Explain URL with general format.* Dec.-13, May-16, Marks 5

1.6 Multipurpose Internet Mail Extension (MIME)

Nov.-11, May-12, 16, Marks 5

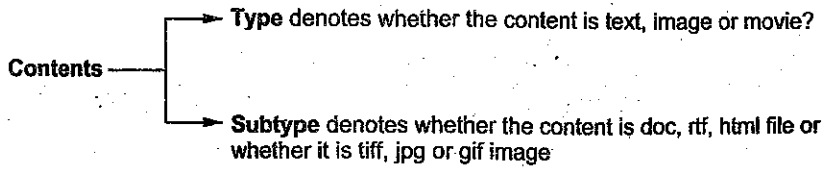
- On the Internet, data is sent in the form of bytes. The receiving software collects these bytes and arranges this data in the proper order. But from these bytes of data it is not clear whether it is text ? Or whether they are a picture ? Or whether they represent a movie ? Then for a user at the receiving end the question arises "How is it possible to know what the data is ?" On the other hand, suppose, in addition to the bytes some additional few bytes are sent along with the actual data telling what the data is then the receiver will easily come to know about the type of data or message. This is what **MIME** does!! It tells what is in a message so that the message contents can be used in an appropriate way.

Definition : *Multipurpose Internet Mail Extensions* (MIME) is an open standard. It specifies how messages must be formatted so that they can be exchanged between different **e-mail systems.**

- MIME is a very flexible format, permitting us to include virtually any type of file or document in an email message.

1.6.1 Type Specification

- The MIME messages can contain text, images, audio, video, or other application-specific data. MIME supports more than 100 different types of content. The content consists of **two levels**



For example : The content of some image can be specified as *Image/jpg*. This means the type of e-mail content is image and subtype is jpg image.

Following table lists various content types used by MIME.

Content Type	Meaning
text/html	HTML text as on the World Wide Web
image/gif	a common image in gif format
audio/midi	midi music format for synthesizers
application/pdf	indicates PDF document

- Example of MIME

```
HTTP/1.0 200 OK
Date: Thu, 14 Dec 2016 00:04:30 GMT
Server: Apache/1.1.0
Content-type: image/jpeg
Set-Cookie: Apache=localhost349860630670983; path=/
Content-length: 32719
Last-modified: Fri, 14 Dec 2016 18:45:59 GMT
```

Here note that the content type is image which is actually a jpeg image.

1.6.2 Experimental Documental types

- Sometimes the experimental subtypes are used.
- The experimental subtype starts with x-
- The web providers can add the experimental subtype in the MIME specification so that particular data can be available to others.

- Supporting web programs or applications called plug-ins are then made available by the web providers.

Review Questions

1. *Explain MIME with its type specification.* Question Bank, Marks 5
2. *What is MIME ? Explain different MIME content types.* Nov. 11, May 16, Marks 5
3. *What is MIME ? Explain briefly MIME type specification and experimental MIME type.* May 12, Marks 5

1.7 The Hypertext Transfer Protocol

Nov. 11, Dec. 16, Marks 5

- Hyper Text Transfer Protocol (HTTP) takes part in web browser and web server communication. Hence it is called a communication protocol. The basic feature of HTTP protocol is that it follows the **request response model**. The client makes a request for desired web page by giving the URL in the address bar. This **request** is submitted to the web server and then web server gives the **response** to the web browser by returning the required web page.

1.7.1 HTTP Request Message Structure

The basic structure of request message is given by following general form –

```
<Start line>
<Header fields>
<Blank Line>
<Message Body>
```

Let us discuss this structure in detail

<Start line>

The **start line** consists of three parts which are separated by a single space. These parts are -

1) Request method 2) Request-URI 3) HTTP version

Let us discuss them in detail.

1. **Request method :** The method defines the CONNECT method which is used during the web browser and server communication. It is always written in Upper Case letters. The primary method in HTTP is GET. The GET method is used when –
   1. You type a URL in address bar.
   2. When you click on some hyperlink which is present in the document.
   3. When browser downloads images for display within a HTML document.

   There is another commonly used method and i.e. POST. The POST method is typically used to send an information collected from a user form. Various methods used by HTTP are as given below -

HTTP Method	Description
GET	This method means retrieve the information requested by the user. The **GET** method is used to retrieve information from a specified URI and is assumed to be a safe, repeatable operation by browsers.
POST	The POST method is used to request the server for desired web page and the request made is accepted as a new subordinate of the resource identified. The **POST** method is used for operations that have side effects and cannot be safely repeated. For example, transferring money from one bank account to another has side effects and should not be repeated without explicit approval by the user.
HEAD	The HEAD method is identical to GET. The only difference is that the server should not return a message-body in the response. The meta-information contained in the HTTP headers in response to a HEAD request should be similar to the information sent in response to a GET request.
OPTION	This method supports for the specified URL. It can be used to check the functionality of a web server by requesting '*' instead of a specific resource.
PUT	This method uploads a representation of the specified resource.
DELETE	This method is useful in deleting the specified resource.
TRACE	When request is made using TRACE method the server echoes back the received request so that a client can see what intermediate servers are adding or changing in the request.

2. **Request URI :** The Uniform Resource Identifier (URI) is a string used to identify the names or resources on the Internet. The URI is a combination of URL and URN. The URL stands for Uniform Resource Locator and URN stands for Uniform Resource

Name. The web address denotes the URL and specific name of the place or a person or item denotes the URN. For example

urn:ISBN 978-81-8431-123-2

specifies the address of some book.

Every URI consists of two parts, the part before the colon : denotes the **scheme** and the part after colon depends upon the scheme. The URIs are case insensitive but generally written in lower case. If the URI is written in the form of http: then it is both an URI and URL but there are some other URI which can also be used as URL. For example

URL	Intended Server
ftp://ftp.mywebsite.com/index.txt	file can be located on FTP server
telnet://mywebsite.org	Telnet server
mailto:myself@mywebsite.org	Mail box
http://www.mywebsite.com	Web server

3. **HTTP version :** The first HTTP version was HTTP/0.9 but the official version of HTTP was HTTP/1.1

Header Fields and message body

The host header filed is associated with the http request. The header fields are in the form of **field name** and **field value.** Thus typical structure of http request is given be following example -

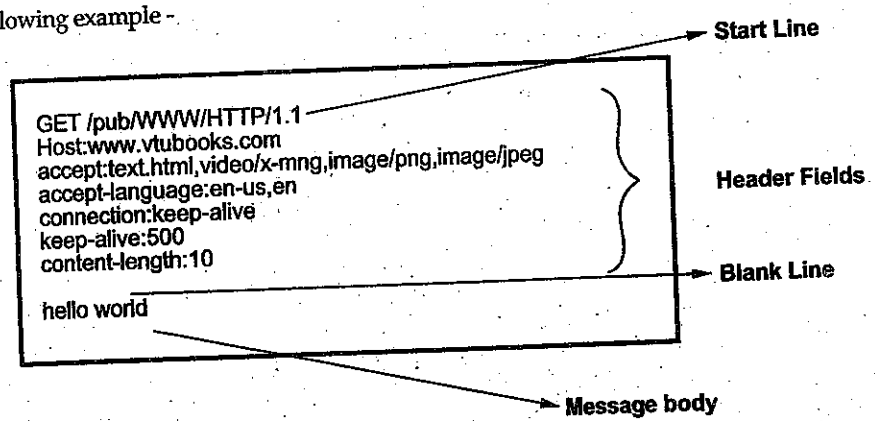


Fig. 1.7.1 HTTP request message structure

1.7.2 HTTP Response Message Structure

The structure of response message is similar to the request message structure. It is as follows

< Status line >
< Header fields >
< Blank Line >
< Message Body >

Let us discuss this structure in detail.

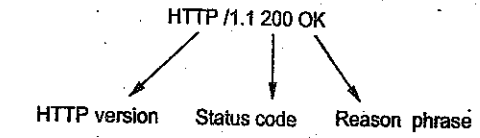
< Status line >

Status line is similar to the start line in the request message. It consists of three fields.

HTTP version	Status code	reason phrase

The HTTP version denotes the HTTP version such as HTTP/1.1. The **status code** is a numeric code indicating the type of response. The **reason phrase** is in the text string form and presents the information about the status code.

For example -



Following table explains some commonly used status codes.

Status Code	Reason Phrase	Description
200	OK	This is a standard response for successful request.
201	Created	It shows that the request is fulfilled and a new resource is being created.
202	Accepted	When the request is accepted for processing but is not processed yet is denoted by this status code.
301	Moved Permanently	The URI for requested resource is moved at some another location.
401	Unauthorized	The requested resource is protected by some password and the user has not provided any password.
403	Forbidden	The requested resource is present on the server but the server is not able to respond it.

404	Not Found	The requested resource is not present currently but may be available in future.
500	Internal Server Error	It is a generic error message that appears due to software internal failure.

< header field > and < message body >

The **header field** in response message is similar to that of request message. The **message body** consists of response message.

For example

```
HTTP/1.1 200 OK
Date: Fri, 1 Jan 2010 07:59:01 GMT
Server: Apache/2.0.50 (Unix) mod_perl/1.99_10 Perl/v5.8.4
mod_ssl/2.0.50 OpenSSL/0.9.7d DAV/2 PHP/4.3.8
Last-Modified: Mon, 23 Feb 2009 08:32:41 GMT
Accept-Ranges: bytes
Content-Length: 2010
Content-Type: text/html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>...</html>
```

Review Questions

1. *Illustrate the HTTP protocol's request and response phases.* Question Bank, Marks 5
2. *Explain the HTTP protocol's request and response phases with an example for each.* Question Bank, Model Question Paper, Marks 10
3. *Explain request phases of HTTP.* Nov.-11, Dec.-16, Marks 5 Dec.-12, 14, Marks 5

1.8 Security

Definition : Web security is the protection of information assets that can be accessed between web server and web client.

- During web client server communication, there are chances that the web browser can illegally share or use the software present on the web server. Similarly web server can make an unauthorized use of information present on the web client.
- If web browsers or web clients have complex software systems then the security problems can be severe.

Security issues

- Following are the most commonly focused **security issues** -

- **Privacy**
  There must be a privacy of data some sensitive information. For example: The web sites for E-purchasing systems must protect the credit card numbers of its clients.
- **Authentication**
  Data on either server or a client must be used by authentic person.
- **Non repudiation**
  The non repudiation is a concept in which the identity of the user must be validated. The user must have to legally validate himself in order to establish a dialogue between client and server.
- **Integrity**
  The information stored on the web must not get modified by some strangers. For example modifying the digital signature, modifying the credit card numbers, modifying the password will spoil the security.

- There are various tools and techniques using which these security issues are maintained. Data encryption and decryption is the most commonly used technique now a day for used for transmitting the message.
- The information which is to be sent over the network is converted into some another form so that nobody else can understand this information. This technique is called **encryption**. And when such information reaches to its destination it is converted back to its original form. This technique is called **decryption**.
- Destructing the data on other computer intentionally or maliciously is another major security problem. This can be done using viruses, worms or Denial Of Services (DOS) attacks. Protection against viruses and worms can be provided by antivirus software and these software must be updated regularly.

Review Questions

1. *Define MIME, Web security, URL.* Question Bank, Marks 5
2. *Discuss web security issues.* Question Bank, Marks 5
3. *What is web security ?* Dec.-12, Marks 5
4. *What is web security ? Explain briefly the security issues.* Dec.-14, Marks 5

1.9 The Web Programmers Toolbox

There are some tools used for programming the web. These tools are nothing but the programming and some scripting/markup languages. Some commonly used markup languages are -

1. XHTML 2. JavaScripts 3. Dynamic HTML 4. XML 5. PHP 6. Perl

Java is a programming language used on both client and server side for web programming.

1.9.1 Overview of XHTML

- XHTML stands for eXtensible Mark-up Language. It is not the programming language but it is a scripting language.
- The purpose of XHTML is to create the layout of the web document that can be displayed on the web browser.
- The XHTML is a mixture of **contents** and **controls.** The **controls** are specified using the **tags** .The contents are the plain text specified within the tags.
- The pair of tags and the contents within them is called the **element**. For example

```
<h2> Welcome to my web page </h2>
```

- Every browser has its own default style of displaying the font, font size, paragraphs and so on. The text will be displayed in this style only. But the developer can modify this display with the help of tags.
- Some tags specify the attributes. The attributes are specified within the double quotes. They are used to provide additional information to the browser. For example

```
<img src="IndiaMap.jpg" />
```

This tag tells the browser to load the image named IndiaMap.jpg

- The latest version of HTML was 4.01 which was replaced by the XHTML1.0 in 2000 by W3C.
- XHTML is very much similar to the HTML 4.01 but XHTML follows the syntax more strictly than the HTML.
- In 2001 the XHTML 1.1 was introduced.

1.9.2 Tools for Creating XHTML Document

- The XHTML document can be written using simple text editor. However there are special XHTML editors available which helps in writing the web document.
- There are two types of XHTML documents that are used more often and those are -
  1. XHTML editor

  2. What you see is what you get- **WYSIWYG** (pronounced as wizzy-wig) XHTML editor.

- XHTML provides some facilities such as spell checking, syntax checking, producing repeated tags(for example putting the buttons on the form), change in color of the text in display to make it easier to read and so on.
- The WYSIWYG XHTML editor is more powerful than the XHTML editors. These text editors are more useful for the beginners and they can write the web document without learning the XHTML. Examples of such text editors are : Microsoft FrontPage, Macromedia Dreamweaver.

1.9.3 Plug-Ins and Filters

- There are two types of converters that are used for XHTML – **plug-ins** and **filters.**
- **Plug-ins** is the set of software components that add the additional capabilities to the XHTML document such as toolbar buttons and menu items.
- The plug-ins are commonly used in web browsers to play video, scan for viruses, and display new file types. Well-known plug-ins examples include **Adobe Flash Player, QuickTime,** and **Microsoft Silverlight.**
- Another kind of converter is **filter.** It converts an existing document in some form into some another form. For example the Microsoft Word can be converted to XHTML.
- Filters are not the part of the editor or word processor. Due to which **platform independency** can be achieved.
- There are **two advantages** of plug-ins and filters
  1. The existing document present in word processor form can be easily converted to XHTML form. Hence user can easily make use of the XHTML document using their word processor.
  2. The XHTML output produced using plug-ins and filters can be easily modified using simple text editor.
- The **disadvantage** of using plug-ins and filters is that if we convert the word processor document to XHTML file then the generated new XHTML file can not be converted back to original form. And the developer has to maintain the new version of the XHTML file.

1.9.4 Overview of XML

- The XML stands for eXtensible Markup Language.
- It is derived from **SGML** i.e. Standard Generalised Mark-up Language.

- The XHTML has **predefined set of tags** on the other hand the XML has its own set of tags and attributes. For example if the user wants to describe the data about the Student then the tags could be <name>, <Roll_no>, <Marks>, <Subject>,<Course> and so on.
- XHTML is used for **representation** on data layout on the web page whereas the XML is used for **description** and **meaning** of data.
- The XML document is validated before processing of data.

1.9.5 Overview of JavaScript

- The JavaScript is a client side scripting language.
- There is **difference between java and JavaScript**. The Java is a programming language whereas JavaScript is a scripting language. The Java must be compiled and interpreted and then the results can be displayed. The JavaScript can be directly displayed on any suitable web browser.
- The main purpose of JavaScript is to validate the form data and to create dynamic XHTML documents.
- The code of JavaScript can be directly embedded within the XHTML document.
- The JavaScript defined the hierarchy of objects. Any XHTML element can be accessed using the objects. Thus objects provide the basis for dynamic documents.

1.9.6 Overview of Java

- Java is originally developed by James Gosling at Sun Microsystems.
- The syntax of this language is derived from C++.
- Java is an object oriented programming language.
- Java provides a specialised code called **byte code** which allows to bring the platform independence. Using this feature the Java code can be executed on any machine.
- Java provides the **applet** which can be directly embedded within the web document. The applets are used for client side programming.
- The server side programming is also possible using Java. The Java **servlets** can be used for that purpose.

1.9.7 Overview of Perl

- CGI i.e. Common Gateway Interface is a standard way by which the server can communicate with the client's browser. That means using CGI the client can

request for some document to the server and the server can return the requested document to the client in his browser.

- The CGI programs are written in **Perl**.
- Perl is highly portable programming language.
- Perl can directly access the **operating system functionalities**, it has a capability of **pattern matching** and support for **database** applications.
- Perl is very powerful and expressive language. It has even replaced C in many applications.
- Perl's syntax is similar to C.
- Perl is compiled to an intermediate language and interpreted.

1.9.8 Overview of PHP

- PHP is a **server side scripting language** specially designed for web applications.
- PHP code can be directly embedded in XHTML document. With PHP, the embedded code of PHP is pre-processed and then the output of PHP is embedded in the XHTML document.
- PHP is similar to JavaScript. It supports the **dynamic nature** of strings and arrays.
- In both PHP and JavaScript, the type of variable is controlled by the value which is assigned to it. This feature is called **dynamic typing**.
- PHP arrays are combination of Perl's arrays and hashes.
- PHP supports **predefined functions for manipulating arrays.**
- PHP allows form handling and database support.

1.9.9 Overview of Ruby

- Ruby got released in 1996 and is designed by **Yukihiro Matsumoto, in Japan.**
- The main characteristic of Ruby is that it is purely an object oriented programming language.
- Every data value in Ruby is **object** and all operations are carried out using the method calls.
- Both the **classes and objects** in Ruby are **dynamic.** That means depending upon the specific object the method can be dynamically added.
- Hence there are **different instances** of objects and methods depending upon the instances at which particular method is invoked.

- The syntax of Ruby is similar to ADA.
- There is no need to declare variable in Ruby. That means it supports the **dynamic typing.**
- In web programming Ruby is used with web development framework **Rails.**

1.9.10 Overview of Rails

- For web application development one framework supported by Ruby is used , which is known as Rails.
- As Rails is bound **closely with Ruby** it is often called as **Ruby on Rails.**
- Rails was developed in 2000 and is released in 2004.
- Rails is based on **Model View Controller (MVC).** This architecture separates presentation and business logic.
- Rails applications are used for relational databases.
- Rails make use of JavaScript framework prototype model to support Ajax.

1.9.11 Overview of Ajax

- Ajax stands for **Asynchronous JavaScript + XML**
- Ajax is useful in development of **interactive web application.**
- In traditional web application interaction, the client requests the server for some document to be displayed on the browser. This request is made by the client by sending the messages to the server. The server then searched for the requested document, processes it and then it is retuned to the client's browser window. This process is time consuming and complicated.
- In Ajax web application there are two changes made from the traditional web interaction process. These are -
  1. The communication from browser to server is **asynchronous.** That means browser does not have to wait for the server to respond. The user of the browser continues his own work after submitting the request to the server. When the server find time it responds to the user.
  2. The server always returns the **document in small part** at a time. So that it takes less time to transmit and render the response.
- Using the **XML support** in Ajax, the data can be displayed in the browser.
- The goal of Ajax is to provide the **efficient web application processing.** So that user can have interactive web application experience.

- Following are some advantages of using Ajax -
  1. The technologies that are used by Ajax are already embedded in web browsers and servers. Hence there is no need to install them separately.
  2. Ajax does not require learning of new tool or language.

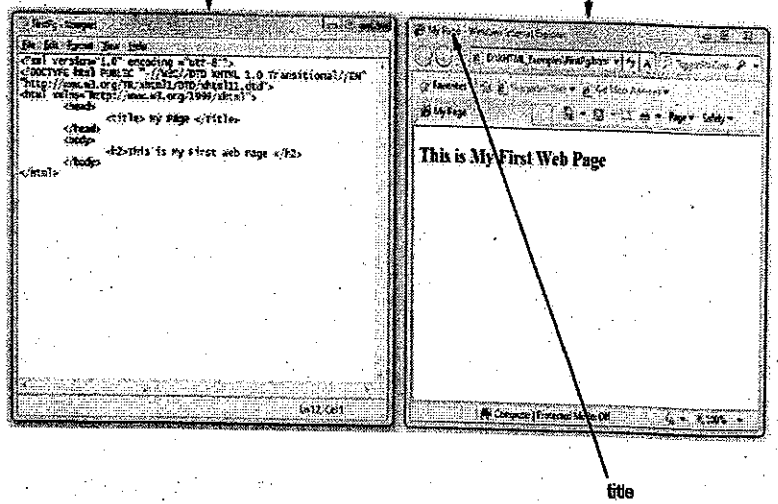
1.10 Introduction to XHTML

May-14, Marks 5

- XHTML stands for eXtensible Hypertext Markup Language. **Hypertext** is simply a piece of text that works as a link.
- **Markup Language** is language of writing layout information within documents.
- The XHTML is a W3C (World Wide Web Consortium) recommendation. Basically an XHTML document is a plain text file and it is very much similar to HTML (i.e. Hyper Text Markup Language). It contains rich text. The rich text means text with tags. Any HTML program can be written in simple Notepad or WordPad text editors. The extension to this program should be either **html** or **htm.** This program then can be opened in some web browser and the corresponding web page can be viewed. Let us create our first web page using HTML.

XHTML document *FirstPg.html* written in Notepad

This XHTML webpage can be viewed in web browser (Internet Explorer). This is an output



- Every XHTML document must begin with an **xml** declaration element that simply identifies the document because it is based on XML. Thus in this line the XML version is specified as 1.0 and encoding as **utf** i.e. the Unicode encoding.
- Note that the XHTML document consists of some strings enclosed within angular brackets. Such strings are called **tags**. Hence various tags that are written in above document are <html>, <h2>, <head> and <body>.
- The XHTML document should be written within **<html>** and **</html>**. The <html> indicates the start of html program and </html> denotes end of html program. Use of slash (/) in the angular bracket indicates end of that particular tag. Note the **xmlns attribute** in <html>, specifies the xml namespace for a document and is required in XHTML documents.

1.10.1 Syntactic Difference between HTML and XHTML

- The XHTML is syntactically identical to HTML. But XHTML follows certain restrictions. These restrictions are mentioned by following difference.

Sr. No.	HTML	XHTML
1.	The HTML tags are **case insensitive**. Hence <body> or <BODY> or <Body> are treated as one and the same.	The XHTML is case sensitive and all the tags in XHTML document must be written in lower case.
2.	We can **omit** the **closing tags** sometimes in HTML document.	For every tag there must be a **closing tag**. Some browsers get confused if the closing tag is not given. There are two ways by which we can mention the closing tags or <a href="Tajmahal.html"/>
3.	In HTML the **attribute values** it not always necessary to **quote** the attribute values. In fact numeric **attribute** values are rarely quoted in HTML. Only if some special characters or white spaces are present in the attribute values then only it is essential to put quotes around them in HTML.	In every XHTML document the attribute values must be quoted.
4.	In HTML there are some implicit attribute values.	In every XHTML the attribute values must be specified explicitly.
5.	In HTML even if we do not follow the nesting rules strictly it does not cause much difference.	In XHTML document the nesting rules must be strictly followed. These nesting rules are : • A form element can not contain another form element. • An anchor element does not contain another form element. • List element can not be nested in the list elements. • If there are two nested elements then the inner element must be enclosed first before closing the outer element. • Text elements can not be directly nested in form elements.

Review Question

1. *Mention the differences between html and xhtml.* **Question Bank, May-14, Marks 5**

❑❑❑

Notes

2 JavaScript and XHTML Document

Syllabus

The JavaScript Execution Environment, The Document Object Model, Element Access in JavaScript, Events & Event Handling, Basic Concepts of Event handling, Events, Attributes & Tags, Handling Events from Body Elements, Handling Events from Button Elements, Handling Events from Textbox & password Elements, The Focus Event, Validating from Input, The DOM 2 Event Model, Event Propagation, Event handler registration, An Example of the DOM 2 Event Model, The Navigator Object, DOM Tree Traversal and Modification, DOM Tree Traversal, DOM Tree Modification.

Contents

2.1 Introduction

- JavaScript was developed by Netscape in 1995. At that time its name was LiveScript. Later on Sun Microsystems joined the Netscape and then they developed LiveScript. And later on its name is changed to JavaScript.
- JavaScript can be effectively used for interaction with the users. The JavaScript support form elements such as button, textbox, menus and so on. Simple web applications such as **calculator, calendar** can be developed using JavaScript.
- Using Document Object Model(DOM) JavaScript can access and modify the properties of CSS(Cascading Style Sheets) and contents of XHTML document. Hence the static web document becomes dynamic.
- JavaScript are used to **validate the data** on the web page before submitting it to the server.

2.2 The JavaScript Execution Environment

- In a browser the XHTML document is displayed. This document is basically displayed in a window. The window is displayed using the **Window** object.
- In JavaScript all the variables are data properties of some object. When we create a global variable then it means we are creating a new property for the **Window** object.
- More than one **Window** objects can be created in JavaScript. And the global variables depend upon the window of its context.
- In JavaScript the **Document** object represents the XHTML document. Every **Window** object has a property named **document** which is reference to the **Document** object.
- **Window** object has a property array **frames** using which multiple frames can be displayed.
- Every **Document** object has a property array called **forms**. Every **Form** object has property arrays called **elements**. These **elements** contain the elements such as buttons and menus.
- The **Document** object also has the property arrays for anchors, links, images and applets.

2.3 The Document Object Model

Nov. 11, May 16, Marks 5

- The Document Object Model provides a mapping for various component of the web document into objects. These objects are accessible by the JavaScript.

- The Document Object Model has been under development by W3C and it is developed in various levels. These levels are

Level	Description
DOM 0	This model is supported by the early browsers. This level could support JavaScript. This version was implemented in Netscape 3.0 and Internet Explorer 3.0 browsers.
DOM 1	This version was issued in 1998 which was focused on XHTML and XML.
DOM 2	This version was issued in 2000 that could specify the style sheet. It also supports the event model and traversal with in the documents.
DOM 3	This is the current release of DOM specification published in 2004. This version could deal with XML with DTD and schema, document validations, document views and formatting.

- Basically DOM is an Application Programming Interface(API) that defines the interface between XHTML document and application program. That means, suppose application program is written in Java and this Java program wants to access the elements of XHTML web document then it is possible by using a set of application programming interfaces(API) which belongs to the DOM.
- The DOM contains the set of interfaces for the document tree node type. These interfaces are similar to the Java or C++ interfaces. They have objects, properties and methods which are useful for respected node type of the web document.

Structure of DOM

- The documents in DOM are represented using a tree like structure in which every element is represented as a node. Hence the tree structure is also referred as **DOM tree.**

For example :

Consider following XHTML document

```
<html>
<head>
    <title>This is My Web Page </title>
</head>
<body>
    <h1>Hello Friends </h1>
    <h2>How are you?</h2>
    <h3>See you</h3>
</body>
```

The DOM tree will be as follows :

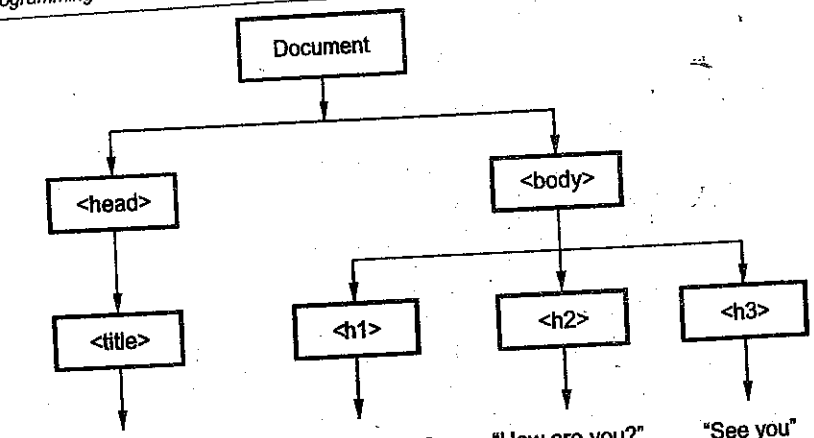


Fig. 2.3.1 DOM structure for simple web document

- DOM node properties
  1. Every element in the DOM tree is called **node.**
  2. The topmost single node in the DOM tree is called the **root.**
  3. Every child node must have a **parent node.**
  4. The bottommost nodes that have no children are called **leaf nodes.**
  5. The nodes that have the common parent are called **siblings.**

Review Questions

1. *Explain the structure of DOM.* — Question Bank, Marks 5
2. *List DOM node properties.* — Question Bank, Nov.-11, Marks 5
3. *What is DOM? Explain.* — Nov.-11, May-16, Marks 5

2.4 Element Access in JavaScript

There are several ways by which we can access the elements of the web document. To understand these methods of accessing we will consider one simple web document as follows

XHTML Document

```
<html>
<head>
    <title>This is My Web Page</title>
</head>
<body>
    <form name="form1">
        <input type="text" name="myinput"/>
</form>
</body>
</html>
```


Method 1 : Every XHTML document element is associated with some address. This address is called **DOM address.** The document has the collection of **forms** and **elements.** Hence we can refer the text box element i.e.

```
<input type = "text" name = "my input"/>
```

Using DOM adress as -

```
var Dom_Obj=document.forms[0].elements[0];
```

But this is not the suitable method of addressing the elements. Because if we change the above script as

```
...
<form name="form1"> <- This is document forms [0]
    <input type="button" name="mybutton"/> <- forms[0].elements[0]
    <input type="text" name="myinput"/> <- forms[0].elements[1]
</form>
...
```

then index reference gets changed. Hence another approach of accessing the elements is developed.

Method 2 : In this method we can make use of the **name** attribute in order to access the desired element.

```
var Dom_Obj=document.form1.myinput;
```

But this may cause a validation problem because the XHTML 1.1 standard does not support the **name** attribute to the form element. Hence another way of accessing is introduced.

Method 3 : We can access the desired element from the web document using JavaScript method **getElementById.** This method is defined in DOM1. The element access can be made as follows –

```
var Dom_Obj=document.getElementById("myinput");
```

But if the element is in particular group, that means if there are certain elements on the form such as radio buttons or check boxes then they normally appear in the groups. Hence to access these elements we make use of its index. Consider the following code sample

```
<form id="Food">
    <input type="checkbox" name="vegetables" value="Spinach" />Spinach
```

```
    <input type="checkbox" name="vegetables" value="FenuGreek" />FenuGreek
    <input type="checkbox" name="vegetables" value="Cabbage" />Cabbage
</form>
```

For getting the values of these checkboxes we can write following code

```
var Dom_obj=document.getElementById("Food");
for(i = 0 ; i < Dom_Obj.vegetables.length ; i++)
document.write(Dom_Obj.vegetables[i]+"<br/>");
```

2.5 Event and Event Handling

- The first specification for **Event Model** is provided by the HTML 4.0 standard.
- This event model is also called as **DOM 0 event model**.
- All the browsers that support the JavaScript also supports the DOM 0 event model.
- Later on the complete and comprehensive event model is named DOM2 is specified.

2.5.1 Basic Concepts of Event Handling

- **Event** is an activity that represents a change in the environment. For example mouse clicks, pressing a particular key of keyboard represent the events.
- **Event handler** is a script that gets executed in response to these events. Thus event handler enables the web document to respond the user activities through the browser window.
- JavaScript support this special type of programming in which events may occur and these events get responded by executing the event handlers. This type of programming is called event-driven programming.
- Events are specified in lowercase letters and these are case-sensitive.
- The process of connecting event handler to an event is called **event registration**. The event handler registration can be done using two methods -
  - Assigning the tag attributes
  - Assigning the handler address to object properties.

2.5.2 Events, Attributes and Tags

On occurrence of events the tag attribute must be assigned with some used defined functionalities. This will help to execute certain action on occurrence of particular event.

Commonly used events and tag attributes are enlisted in the following table -

Events	Tag Attribute	Meaning	Associated Tags
blur	onblur	losing the focus	<button> <input> <a> <textarea> <select>
change	onchange	on occurrence of some change	<input> <textarea> <select>
click	onclick	When user clicks the mouse button	<a> <input>
dblclick	ondblclick	When user double clicks the mouse button	<a> <input> <button>
focus	onfocus	When user acquires the input focus	<a> <input> <select> <textarea>
keyup	onkeyup	When user releases the key from the keyboard	form elements such as input,button,text,textarea and so on
keydown	onkeydown	When user presses the key down	form elements such as input,button,text,textarea and so on
keypress	onkeypress	When user presses the key	form elements such as input,button,text,textarea and so on
mousedown	onmousedown	When user clicks the left mouse button	form elements such as input,button,text,textarea and so on
mouseup	onmouseup	When user releases the left mouse button	form elements such as input,button,text,textarea and so on
mousemove	onmousemove	When user moves the mouse	form elements such as input,button,text,textarea and so on
mouseout	onmouseout	When the user moves the mouse away from some element	form elements such as input,button,text,textarea and so on
mouseover	onmouseover	When the user moves the mouse away over some element	form elements such as input,button,text,textarea and so on
load	onload	After getting the document loaded	<body>
reset	onreset	When the reset button is clicked	<form>

submit	onsubmit	When the submit button is clicked	<form>
select	onselect	On selection	<input> <textarea>
unload	onunload	When user exits the document	<body>

The use of these tag attributes for handling the events is illustrated by following code sample

```
<input type = "button" name="My_button"
        onclick="My_fun()"; />
```

Tag attribute — Event handler

That means when the user clicks the **button** then as an event handler a user defined function **My_fun()** gets called. Basically in this function user writes the instructions that need to be executed on the button click event.

Review Question

1. *Define event and event handling* **Question Bank, Model Question Paper. Marks 2**

2.6 Handling Events from Body Elements

- For handling the event from body element the most commonly used events are **onload** and **unload**.
- The **onload** event gets activated as soon as the web page gets loaded in browser's window.
- **Example :**

JavaScript[OnloadDemo.html]

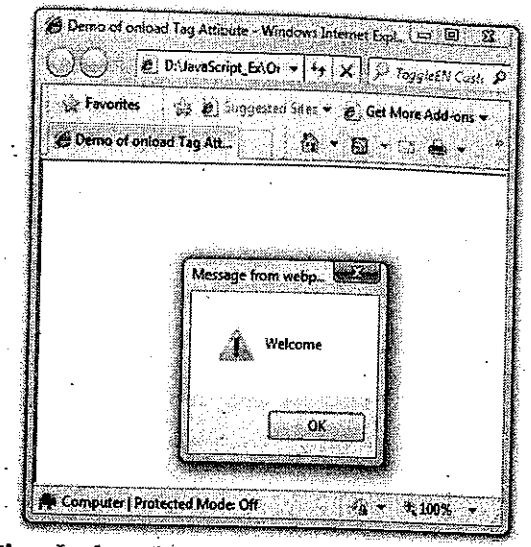
```
<!DOCTYPE html>
<html>
<head>
 <title>Demo of onload Tag Attribute</title>
 <script type = "text/javascript">
 function my_fun()
  {
 //This message will be displayed on page loading
   alert("Welcome");
```



```
 }
 </script>
</head>
<body onload="my_fun()">
</body>
</html>
```

When web document gets loaded on the browser window then my_fun() will be called

Output



Script Explanation : In above JavaScript,

- For **the body** element the **onload** event is triggered and this onload event calls a function **my_fun()**
- Inside **the my_fun()** function the alert box displaying *Welcome* message is specified.
- Hence as **soon** as we load the above web page we get a welcome message.

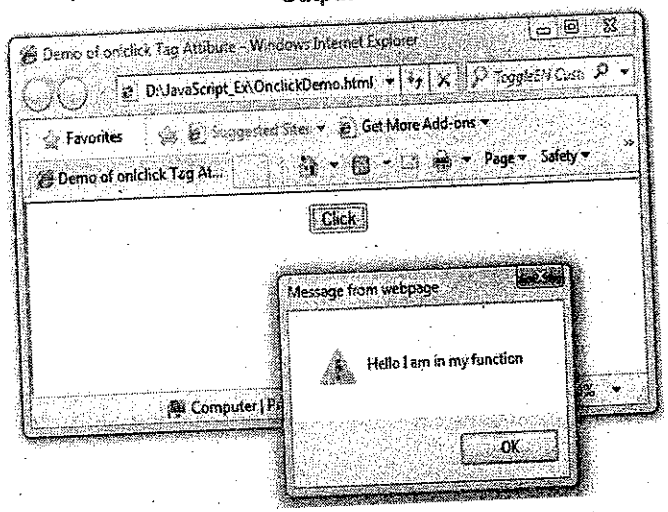
2.7 Handling Events from Button Elements

- For **handling** the event using button element we have used the tag attribute **onclick.**
- The idea **is that** whenever we click the button some event handler must be called. This **event handler** can be a **user defined function** in which certain set of instructions **get** executed.
- **Example 1 :** Following is a simple JavaScript in which on the button click we have **called a function my_fun().** This is a simple function in which we have displayed **some** message using alert popup box.

JavaScript [OnclickDemo.html]

```
<!DOCTYPE html>
<html>
<head>
 <title>Demo of onlclick Tag Attibute</title>
<script type="text/javascript">
 function my_fun()
 {
  alert("Hello I am in my function");
 }
</script>
</head>
<body>
 <center>
 <form>
   <input type="button" value="Click" onclick="my_fun()">
 </form>
 </center>
 </body>
 </html>
```

Output



Now let us see another example

- **Example 2 :**

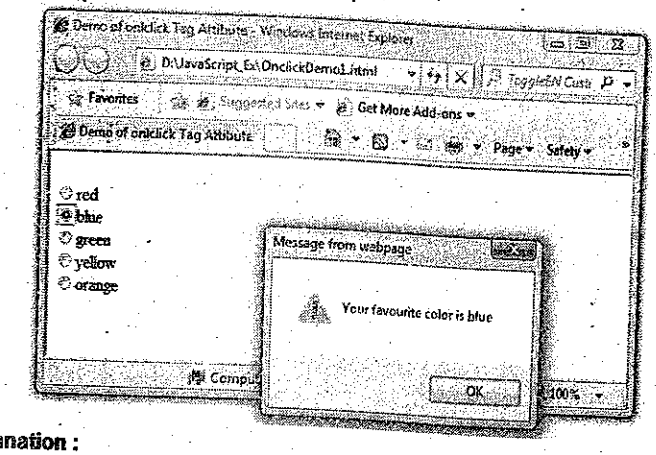
JavaScript[OnclickDemo1.html]

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



```
 <title>Demo of onlclick Tag Attibute</title>
<script type="text/javascript">
function fun(form_obj)
{
 alert("Your favourite color is "+form_obj.value);
}
</script>
</head>
<body>
<center>
<form>
   <div align="left"><br/>
     <input type="radio" name="group1" value="red" onclick=fun(this)>red<br/>
     <input type="radio" name="group1" value="blue" onclick=fun(this)>blue<br/>
     <input type="radio" name="group1" value="green" onclick=fun(this)>green<br/>
     <input type="radio" name="group1" value="yellow" onclick=fun(this)>yellow<br/>
     <input type="radio" name="group1" value="orange" onclick=fun(this)>orange<br/>
   </div>
</form>
</center>
</body>
</html>
```

Output



Script explanation :

- In above script the event handler is registered by assigning its call to **onclick** attribute of radio buttons.
- The specific button that is clicked is identified by sending the parameter to the event handler function fun.

- We have passed the corresponding object as the specific parameter using **this** pointer.
- In the event handler routine using the value of sent object the particular button click can be identified.
- **Example 3 :**
- We can modify the above script in order to access the element by using the DOM object. This DOM object can be obtained using the method **getElementById**

JavaScript[OnclickDemo1.html]

```
<!DOCTYPE html>
<html >
<head>
 <title>Demo of onlclick Tag Attibute</title>
 <script type="text/javascript">
  function fun(choice)
  {
   var Dom_obj=document.getElementById("form1");
   for(var i=0;i<Dom_obj.group1.length;i++)
    if(Dom_obj.group1[i].checked)
    {
     choice=Dom_obj.group1[i].value;
     break;
    }
   switch(choice)
   {
    case "red": alert("Your favourite color is "+choice);
          break;
    case "blue": alert("Your favourite color is "+choice);
          break;
    case "green": alert("Your favourite color is "+choice);
          break;
    case "yellow": alert("Your favourite color is "+choice);
          break;
    case "orange": alert("Your favourite color is "+choice);
          break;
   }
  }
 </script>
</head>
<body>
<center>
<form id="form1">
 <div align="left"><br/>
```

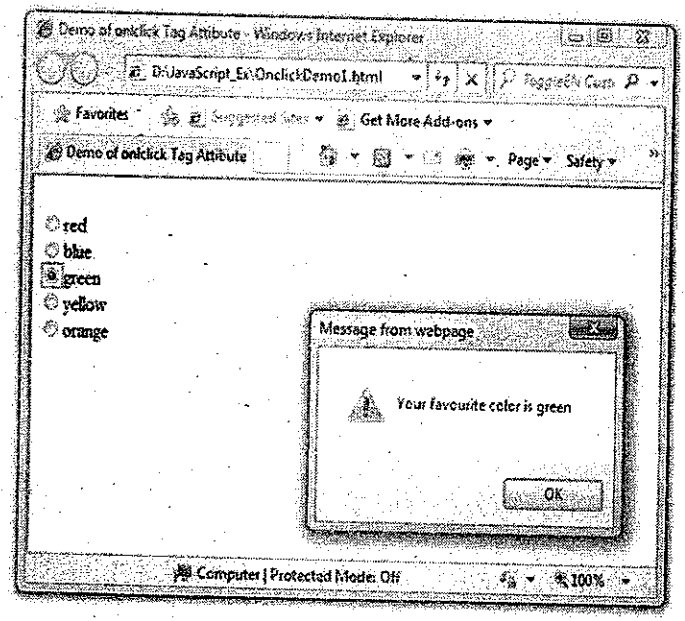
Function definition

Displaying appropriate color messages


```
    <input type="radio" name="group1" value="red" />red<br/>
    <input type="radio" name="group1" value="blue" />blue<br/>
    <input type="radio" name="group1" value="green" />green<br/>
    <input type="radio" name="group1" value="yellow" />yellow<br/>
    <input type="radio" name="group1" value="orange" />orange
  </div>
 </form>
 <script type="text/javascript">
  var Dom_obj=document.getElementById("form1");
  Dom_obj.elements[0].onclick=fun;
  Dom_obj.elements[1].onclick=fun;
  Dom_obj.elements[2].onclick=fun;
  Dom_obj.elements[3].onclick=fun;
  Dom_obj.elements[4].onclick=fun;
 </script>
</center>
</body>
</html>
```

Call to the function onclick. That means registering the event

Output



Just execute the above script, select different radio buttons and accordingly you will get the appropriate message using the alert box.

2.8 Handling Events from Text Box and Password Elements

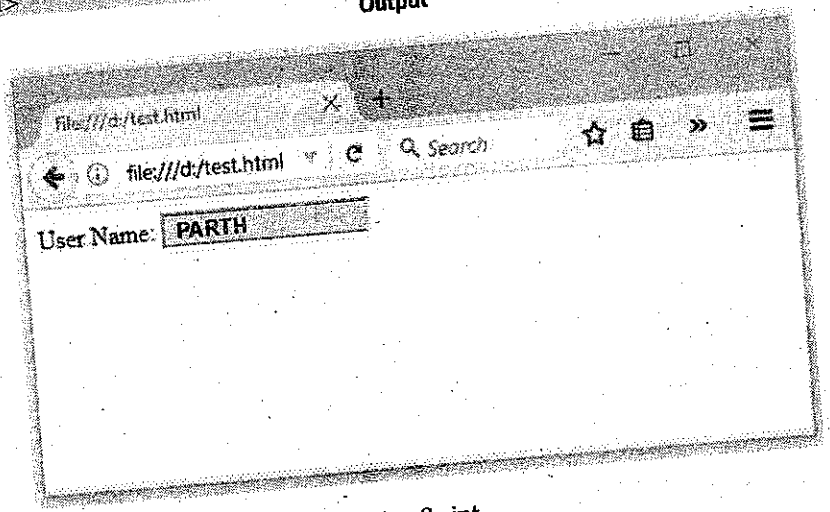
May 13, Marks 10

2.8.1 The Focus Event

- The **onfocus** event occurs when particular element need to be focused.
- The **onfocus** event is exactly opposite to **onblur** event.
- This event is generally applied with the elements <input>, <select> and <a>
- **Example**

```
<!DOCTYPE html>
<html>
<body>
   <script>
   function my_fun(obj)
   {
     obj.style.background = "blue";
   }
   </script>
   User Name: <input type="text" onfocus="my_fun(this)">
</body>
</html>
```

Output



Script Explanation : In above given JavaScript,

We have used **focus** event with **<input>** element. Hence the textbox gets a focus.

This change in focus can be represented by changing the background color of the text box. Using the **style.background** attribute we can set any desired background color of the text box.

2.8.2 Validating from Input

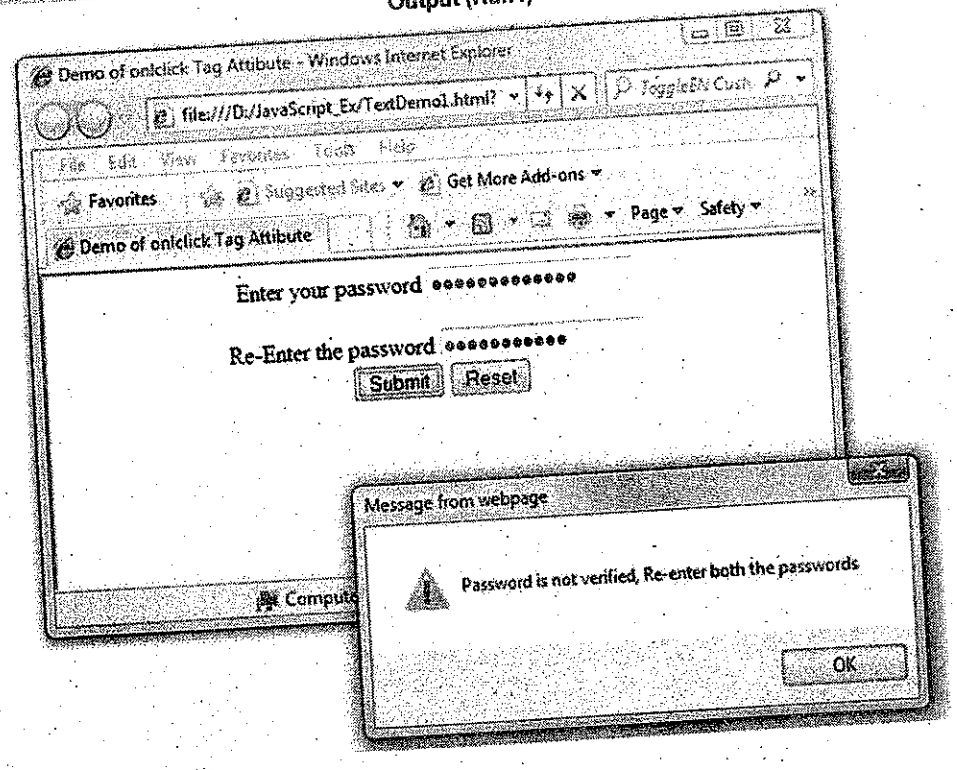
- We have normaly used password verification process that comes along the web document.
- In this process user has to enter the password correctly for two times. If both the passwords are matching then the password is verified.
- Normally this facility is given when user creates his web account.
- In the following JavaScript we have used two textboxes which are of **password** type. That means whatever we type in these boxes appear in the form of dots. We will compare the entries in the two text boxes; if those are not same then the alert message will be displayed.

JavaScript[TextDemo.html]

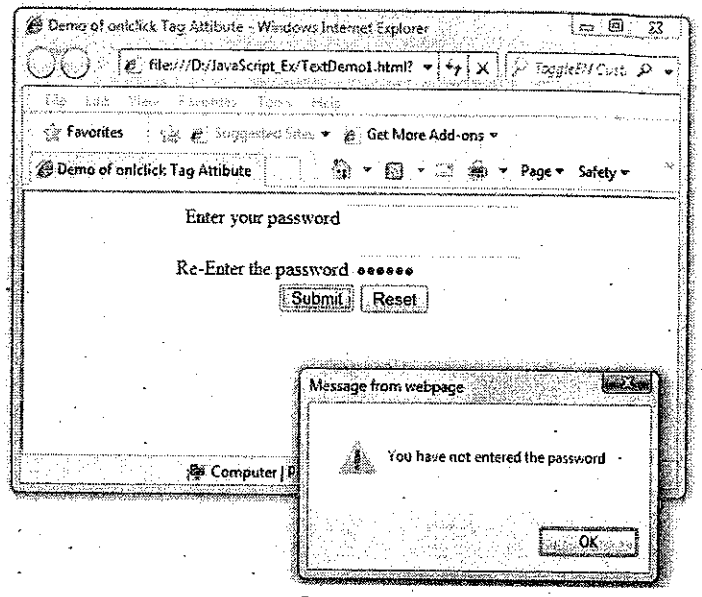
```
<!DOCTYPE html>
<html>
<head>
 <title>Demo of onlclick Tag Attibute</title>
<script type="text/javascript">
function my_fun()
{
 var mypwd=document.getElementById("pwd");
 var my_re_pwd=document.getElementById("re_pwd");
 if(mypwd.value=="")
 {
   alert("You have not entered the password");
   mypwd.focus();
   return false;
 }
 if(mypwd.value!=my_re_pwd.value)
 {
   alert("Password is not verified, Re-enter both the passwords");
   mypwd.focus();
   mypwd.select();
   return false;
 }
 else
 {
   alert("Congratulations!!");
   return true;
 }
}
</script>
</head>
<body>
 <center>
```

```
<form id="form1">
 <label> Enter your password
 <input type="password" value="" id="pwd" />
 </label>
 <br/><br/>
 <label> Re-Enter the password
 <input type="password" value="" id="re_pwd" onblur="my_fun();"/>
 </label><br/>
 <input type="submit" value="Submit" name="submit" onsubmit="my_fun();"/>
 <input type="reset" value="Reset" name="reset"/><br/>
</form>
</center>
</body>
</html>
```

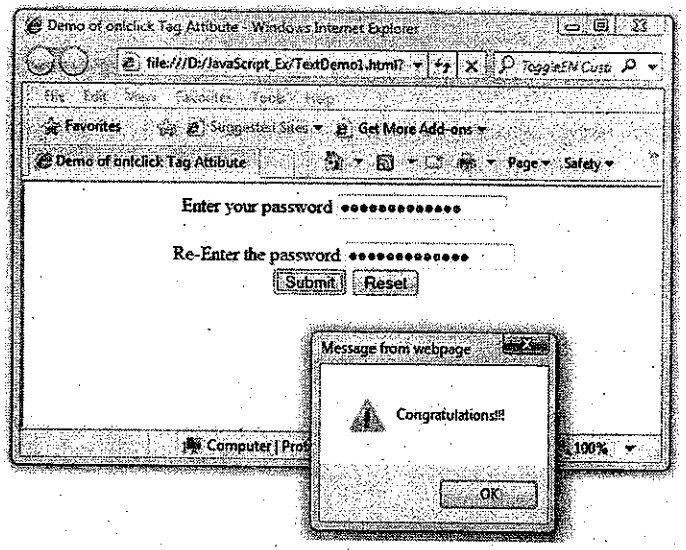
Output (Run1)



Output (Run2)



Output (Run3)



Review Question

1. *Explain how to handle the focus event with an example* **Question Bank, May-13, Marks 10**

2.9 The DOM 2 Event Model

- The DOM2 event model provides the generic event system in which the events can be registered.
- The event flow can be represented in a hierarchical manner.
- Using DOM2 event model information of each event can be obtained.
- The DOM2 provides the method for capturing the events so that one can perform specific action in response to them.
- It also provides an **Event** and **MouseEvent** objects. These are basically the interfaces which contain information specific to a given event. This information can then be used by your event processing code.
- Following table enlists the event interface and types of various events that can be triggered.

Event interface	Various events
Event	blur,change,abort,error,focus,load,reset,resize,select,scroll,unload,submit
MouseEvent	mousedown, mouseup, mouseover, mousemove, mouseout, click

- In DOM 0 the connection between even and event handler is simple. But in DOM 2 the event model is complicated one.

2.9.1 Event Propagation

- Event propagation means travelling of an event. To understand the event propagation consider one scenario -

  Suppose there is an element inside other element. And both these elements have **onchange** event handlers. Then the question occurs here is that which event will be handled first whether the event in the outer element or the event in inner element?. Unfortunately there are two opinions about it - the first opinion is the event in the outer element should be handled first and then the event in the inner element should get handled(*event capturing*).

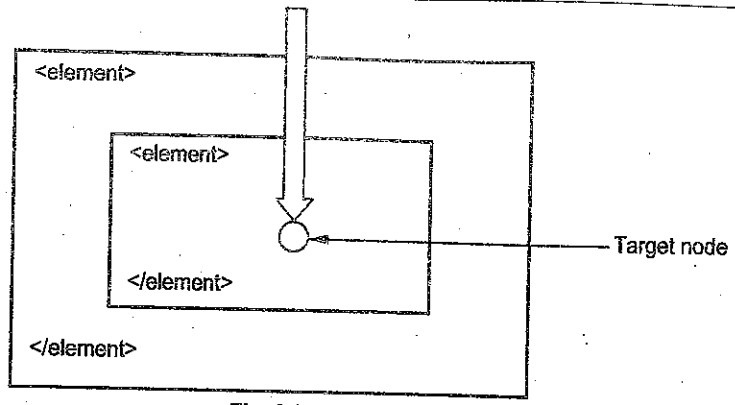


Fig. 2.9.1 Event capturing

- According to other opinion, first directly the event which is present in the inner element should be handled first and then the event in the outer element must be handled(*event bubbling*). The DOM 0 supports this kind of event handling.

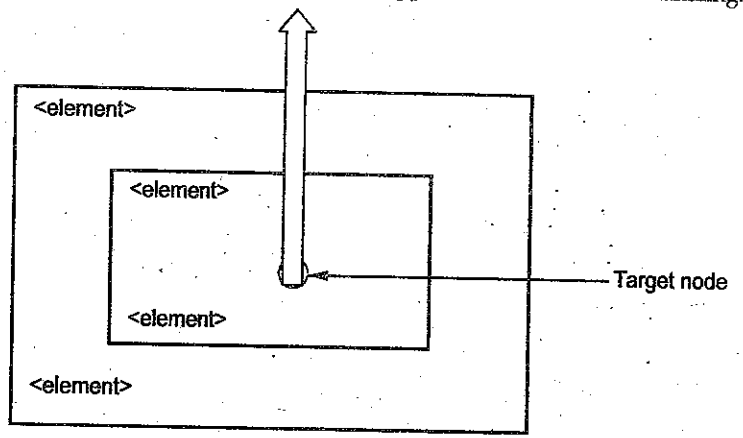


Fig. 2.9.2 Event bubbling

- Event object is created at a node in the document tree. This node is called the **target node**
- To solve the controversy of event handling whether it is using capturing or using bubbling, the W3C suggested the **DOM2 event model** in which both the approaches are considered.
- According to DOM2 event model the event handling is a three-phase process

Phase 1

This phase is called the **capturing phase** in which the event starts at root node in the document tree and propagates towards the target node. In this direction if any event

handler is found(not the target node event handler) then first of all it is checked whether they are enabled or not. If there is any enabled event then it is executed by the capturing phase(may or may not be enabled event). Thus event then reaches to the target node

Phase 2

In this phase the handlers registered to the target node get executed.

Phase 3

This phase is called the **bubbling phase** in which the event starts at the target node and moves up to the document tree node. In this phase the event bubbles up the document tree to the root node. On this trip if any event handler registered for event gets executed by the bubbling phase(may or may not be enabled event).

- There are some events that can be bubbled or captured. All the mouse events can be bubbled but load and unload events do not get bubbled.
- The **stopPropagation** method of event object is useful for stopping the propagation of an event.
- Events cause certain action on occurrence of particular action. Using **preventDefault** method the default action can be prevented on occurrence of an event.

2.9.2 Event Handler Registration

- The event handler registration in DOM0 is called the traditional method of event registration. In DOM0 there are **two methods** by which the event registration is done -

1) By assigning event handler script to "event-tag attribute"

- **For example :**

```
< input type = "button" onclick = "alert("Hello");"/>
```

If there are more than one statement to be executed then we can use a function.

- **For example :**

```
< input type = "button" onclick = "my_function ();"/>
```

2) By assigning the address of handler function to "event property" of object which is associated with HTML element.

- **For example :**

```
Document.getElement By Id ("form1"). Onclick = my_function;
```

- **Drawback** The drawback of using DOM0 event model for registering the event is that we cannot assigned two different handler for the same event. Sometimes we

want two different functionalities to get executed for the same type of event. For instance we may want to check the spelling on button click or we may want to edit the code on button click. This can not be achieved in DOM0 event registration model.

- To overcome the drawback of traditional event registration model W3C has introduced the new mode called DOM2 event registration model. That means at a time we can have more than one event handler functions that can be associated with the same event.
- In this model a method **addEventListener** is called to which the name of the event and event listener is passed as a parameter.
- The **addEventListener** is defined in **EventTarget** interface.

```
addEventListener("name_of_event",handler_function,true or false);
```

The first parameter is the name of the event which must be given in quotes as it is considered to be the string literal. The second parameter is the event handler function which is basically the user defined function for specifying the actions to be taken when the event gets fired. The third parameter is either true or false. If the value is **true** then that means the particular event is enabled for using it in **capturing phase**. If the value is **false** then that means the particular event is enabled for using it in **bubbling phase**

- **For example -**

```
document.my_button.addEventListener("click",my_fun,false);
```

When user clicks the button element whose id is **my_button** the event handler code **my_fun** gets invoked. The third parameter **false** indicates that the event is enabled for the bubbling phase. Thus addEventListener is useful in registering the event.

For removing or deleting the event registration the method **removeEventListener** is used. This method takes the same parameters as the **addEventListener**. Hence

```
removeEventListener("name_of_event",handler_function,true or false);
```

- The **this** keyword is used to refer the owner of a function. In event registration function if we use the **this** keyword then it refers to the element the event is handled by

For example -

```
document.my_button.addEventListener("click",my_fun,false);
```

The event handler function **my_fun** will be

```
function my_fun()
{
    this.style.background = "pink";
}
```

That means if we click the button my_button then the background color gets changed to pink.

- There is a event property of global window object which provides the event details.

For example

```
<input type="text" value=" "id="my_txt" />

Txt_obj=getElementById("my_txt");
Txt_obj.addEventListener("change",my_fun,false);

function my_fun(event)
{
 var My_name=event.currentTarget;
 alert(My_name.value);
}
```

Placing the text box on the form

Registering the event when user types something in the text box

The contents that are typed by the user in the text box gets displayed using alert

Using the currentTarget property of the event object the target node of the event can be accessed.

- The **clientX and clientY** given the x and y co-ordinates at the time of event.

2.9.3 An Example of DOM 2 Event Model

In this section we will make use of DOM 2 event model for registering the event. Note that we have used two different functions for the same event.

Note that this feature is not supported by Internet Explorer. Hence we must see the output on some another web browser. I have used Mozilla Firefox web browser.

JavaScript [EventReg.html]

```
<!DOCTYPE html>
<html>
<head>
 <title>Demo of Event Registration</title>
<script type="text/javascript">
 function chkDt(event)

 {
 var inputstr =event.currentTarget;
 var str=inputstr.value;
 if(str.match(/\d{1,2}\.\d{1,2}\.\d{4}/))
 {
```

Accessing the text box created for entering the name using the **event.currentTarget**


```
 a=str.split(".");
 var dd=Number(a[0]);
 var mm=Number(a[1]);
 var yy=Number(a[2]);
 if(mm==1||mm==01)
 {
 if(dd>=30)
 {
 alert("Invalid date,Please re-enter it(February have 28 or 29 days)");
 inputstr.focus();
 inputstr.select();
 }
 else
 alert("Valid Date");
 }
 else if(mm==0||mm==2||mm==4||mm==6||mm==7||mm==9||mm==11)
 {
 if(dd>31)
 {
 alert("Invalid date,Please re-enter it");
 inputstr.focus();
 inputstr.select();
 }
 else
 alert("Valid Date");
 }
 else if(mm==3||mm==5||mm==8||mm==10)
 {
 if(dd>30)
 {
 alert("Invalid date,Please re-enter it");
 inputstr.focus();
 inputstr.select();
 }
 else
 alert("Valid Date");
 }
}
 else
 {
 alert("invalid Input format!!,Please try again");
 inputstr.focus();
 inputstr.select();
 }
}
function chkName(event)
```

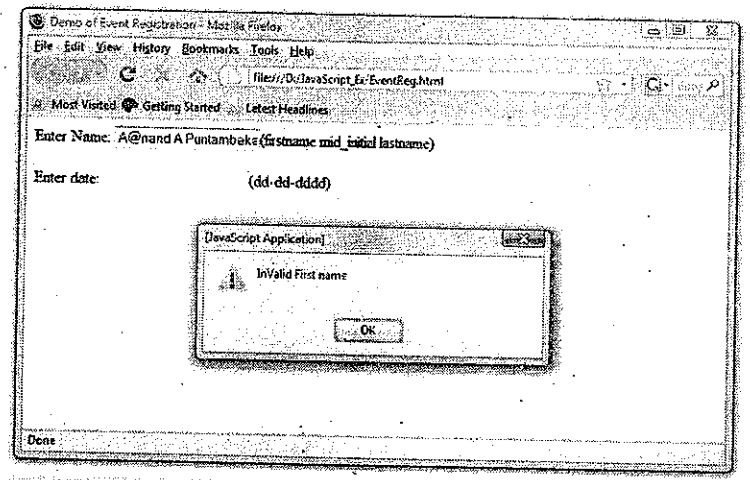
```
{
var inputstr =event.currentTarget;
var str=inputstr.value;
if(str.match(/[a-zA-Z]+\s[a-zA-Z]\s[a-zA-Z]+/))
{
 a=str.split(" ");
 var First_name=a[0];
 var Mid_initial=a[1];
 var Last_name=a[2];
 if(First_name.match(/^[a-zA-Z]{1,15}$/))
 {
  if(Mid_initial.match(/[a-zA-Z]$/))
  {
   if(Last_name.match(/^[a-zA-Z]{1,15}$/))
    alert("Valid name, now enter date");
   else
    alert("InValid Last Name");
  }
  else
   alert("InValid Middle name initial");
 }
 else
  alert("InValid First name");
}
else
  alert("You have entered wroug input");
}
</script>
</head>
<body>
 <form id="form1">
  <label>Enter Name:
   <input type="text" value="" id="Name_chk" />(firstname mid_initial lastname)
  </label>
  <br/><br/>
  <label>Enter date:
   <input type="text" value="" id="Dt_chk" />(dd-dd-dddd)
  </label>
  <br/>
 </form>
<script type="text/javascript">
```

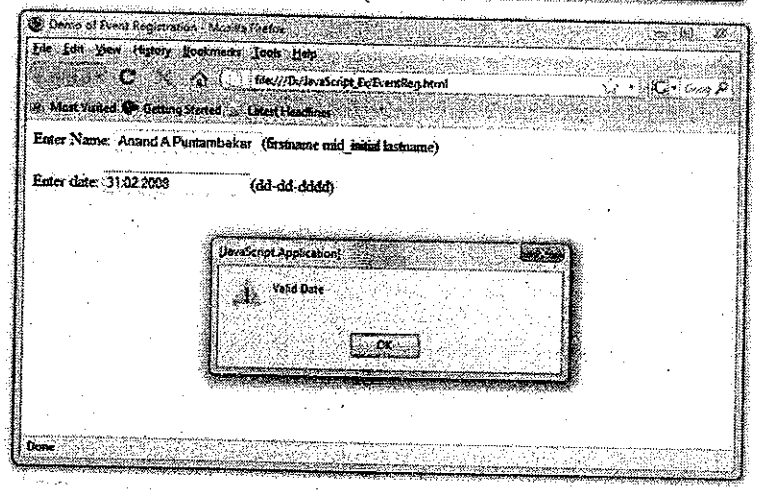
Accessing the text box created for entering the name using the event.currentTarget


```
var name_node=document.getElementById("Name_chk");
var dt_node=document.getElementById("Dt_chk");
dt_node.addEventListener("change",chkDt,false);
name_node.addEventListener("change",chkName,false);
</script>
</body>
</html>
```

Getting the object for each element and then registering the event handlers

Output





Review Questions

1. *Discuss the two ways to register an event handler in DOM-0 event model.*

Question Bank, Marks 10

2. *Explain the 3 phases of event processing in the DOM-2 event model*

Question Bank, May 12, Marks 10

Nov. 11, Marks 5

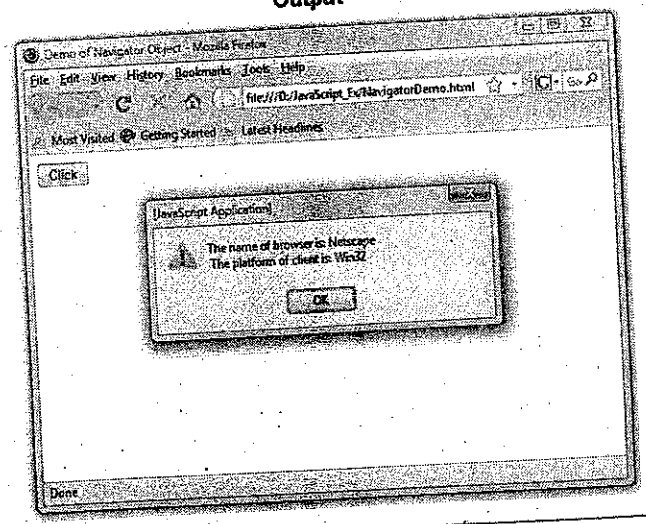
2.10 The Navigator Object

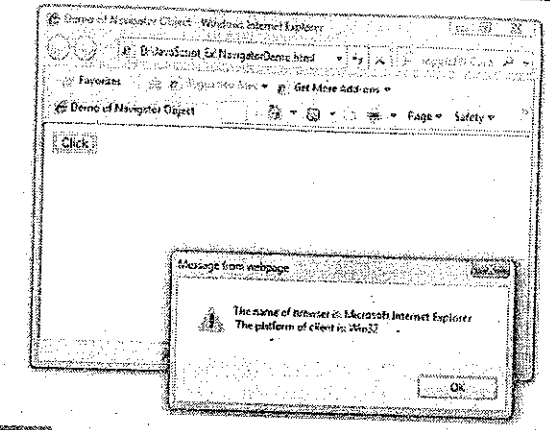
The navigator object is the object of JavaScript which is used to get certain kind of information such as name of the browser, its version, the platform on which the client is working, the language of the browser and so on. Following JavaScript makes use of navigator object -

JavaScript[navigatorDemo.html]

```
<!DOCTYPE html>
<html>
<head>
<title>Demo of Navigator Object</title>
<script type="text/javascript">
function my_fun()
{
alert("The name of browser is: "+navigator.appName +"\n"+" The platform of client is: "+navigator.platform);
}
</script>
</head>
<body>
<form>
<input type="button" value="Click" onclick="my_fun();" />
</form>
</body>
```

Output





Review Question

1. *Write a note of navigator object.* Nov. 11, Marks 5

2.11 DOM Tree Traversal and Modification

For traversing the DOM tree there is an object **Element** which can have various properties using which the Document tree can be traversed. In other words suppose we have written some XHTML web document and if we want to check - what are the elements in that web document or if we want to check which is the first element in that web document then such queries can be solved by allowing the designer to walkthrough the web document. Similarly particular element can also be modified.

2.11.1 DOM Tree Traversal

Addressing XHTML using forms and elements

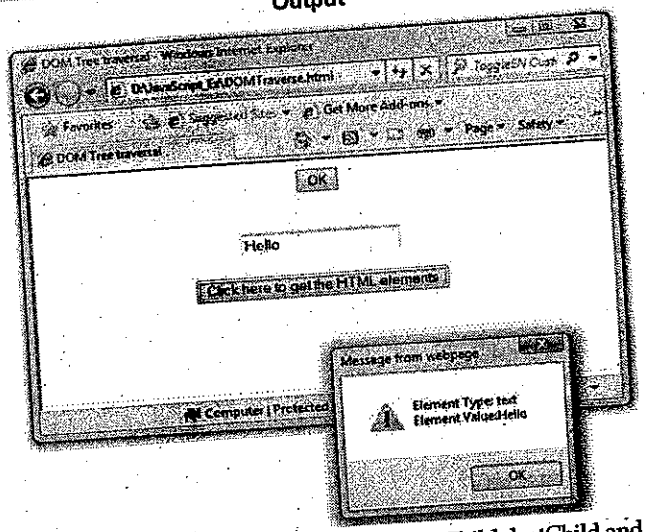
- For traversing such tree simple JavaScript can be written.
- Following is an illustration in which we have used a form object over which we have placed three elements such as button, text box and submit button. As an output we can obtain of these elements.

JavaScript

```
<!DOCTYPE html >
<html>
<head>
<title>DOM Tree traversal</title>
<script type="text/javascript">
function my_fun()
{
var Dom_obj=document.getElementById("form1");
```

```
alert(Dom_obj.nodeName);
alert("There are "+Dom_obj.length+" elements of form element and those are ");
for(i=0;i<Dom_obj.length;i++)
    alert("Element Type: "+Dom_obj.elements[i].type+"\n"+"Element
    Value:"+Dom_obj.elements[i].value);
}
</script>
</head>
<body>
<center>
<form id="form1">
 <input type="button" value="OK"/><br/>
<br/><br/>
 <input type="text" value="Hello" id="mytext"/>
<br/><br/>
 <input type="submit" value="Click here to get the HTML elements" onclick="my_fun();"/>
 </form>
<p><em>

</em></p>
</center>
</body>
</html>
```

Output



We can use various properties such as nextSibling, firstChild, lastChild and so on in order to traverse through the DOM tree. In above given script we have used simple properties such as **nodeName, value and type** to access the particular node.

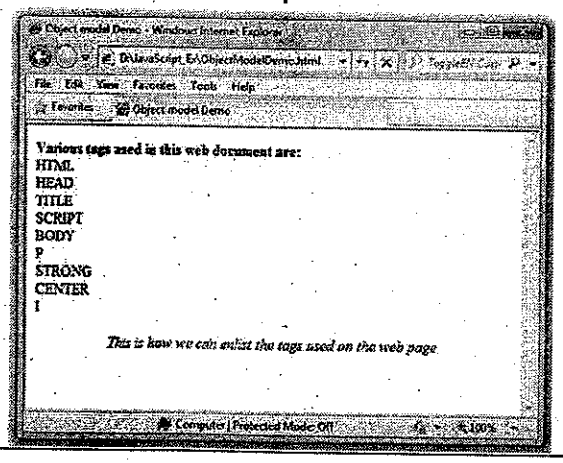
Use of object all

- There is a special object model collection called all which is used to refer all the HTML elements.
- If we want to represent all the HTML elements then the object model collection **all** is used.
- The order in which these HTML elements come in our program in the same order all those elements will be displayed.
- In the following program we have used **all** in order to display all the HTML tags used by our program.

```
<html>
<head>
<title>Object model Demo</title>
<script type="text/javascript">
var web_page_element="";
function Display()
{
for(i=0;i<document.all.length;i++)
 web_page_element+="<br>"+document.all[i].tagName;
pmsg.innerHTML+=web_page_element;
}
</script>
</head>

<body onload="Display()">
<p id="pmsg"><strong>Various tags used in this web document are:</strong></p>
<center><i>This is how we can enlist the tags used on the web page</i></center>
</body>
</html>
```

The object **all** belongs to the **document**. And its **tagName** property helps to find the name of HTML tag.

Output



2.11.2 DOM Tree Modification

We can also the existing structure of DOM tree by removing some child or replacing one child by another child. For that purpose **replaceChild** property is used.

We can use **insertBefore** method for placing the new child node before the referred child.

The **appendChild** method is used for inserting new child at the end of the listing of siblings. ❑❑❑

3 Dynamic Documents with JavaScript

Syllabus

Introduction, Positioning Elements, Absolute Positioning, Relative Positioning, Static Positioning, Moving Elements, Element Visibility, Changing Colors & Fonts, Changing Colors, Changing Fonts, Dynamic Contents, Stacking Elements, Locating the Mouse Cursor, Reacting to the Mouse Click, Slow Movement of Elements, Dragging & Dropping Elements.

Contents

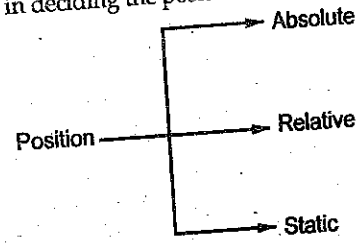
3.1 Introduction

- The first important difference between HTML and DHTML is that HTML is used to create **static web pages** and DHTML is used to create **dynamic web pages.**
- The HTML is consists of simple HTML tags, on the other hand DHTML is made up of XHTML tags + Cascading Style Sheets(CSS) + JavaScripts.
- Creation of HTML web pages is **simplest** but less interactive. Creation of DHTML is **complex but more interactive.**

3.2 Positioning the Elements

Nov.-11, Dec.-13, Marks 10

- Using dynamic HTML web designer have a full control over positioning the elements. There is a **position** property. There are some style properties **left** and **top** which are used in deciding the position of an element on the web document.



3.2.1 Absolute Positioning

Definition : This is a type of positioning that allows to actually place any page element exactly where the developer wants it. We use the positioning attributes top, left bottom and right to set the location.

Following is an illustration positioning the element using absolute positioning.

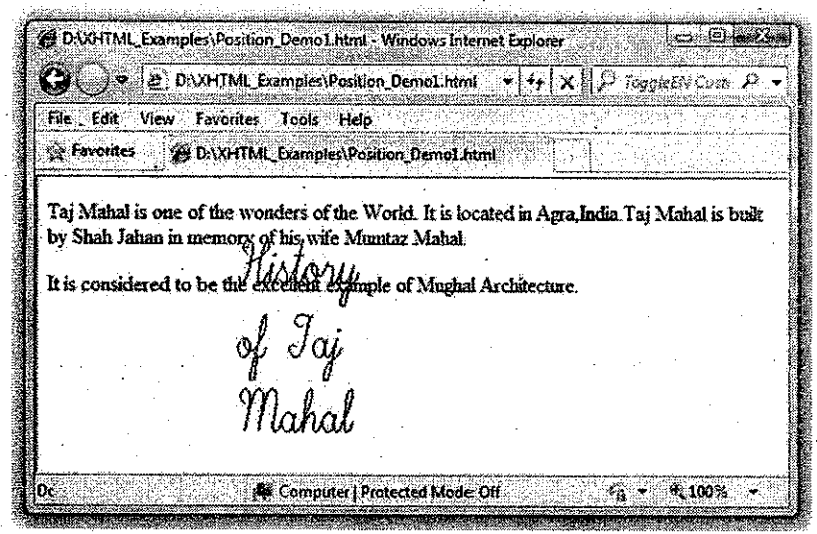
DHTML document[Position_Demo1.html]

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <style type="text/css">

  .text_pos
  {
  position:absolute;
  left:150px;
```



```
  top:-10px;
  font-family:script;
  font-size:50px;
  width:100px;
  }
</style>
</head>
<body>
<p>Taj Mahal is one of the wonders of the World. It is located in Agra,India.Taj Mahal is built by Shah Jahan in memory of his wife Mumtaz Mahal.
</p>
<p>It is considered to be the excellent example of Mughal Architecture.</p>
<p class="text_pos">History of Taj Mahal</p>
</body>
</html>
```

Output



3.2.2 Relative Positioning

Definition : If we assign the value to the position attribute as **relative** then the corresponding element gets placed at the relative position to the previous contents.

- For instance if there is a paragraph which is starting from leftmost and topmost position in the browser's window and if we want to place the element <h1> with top:70px then <h1> will be displayed 70 pixels from paragraph.
- Thus relative positions get decided with respect to the previous contents.

Following is an example in which relative positioning is used.

DHTML Document[PositionDemo2.html]

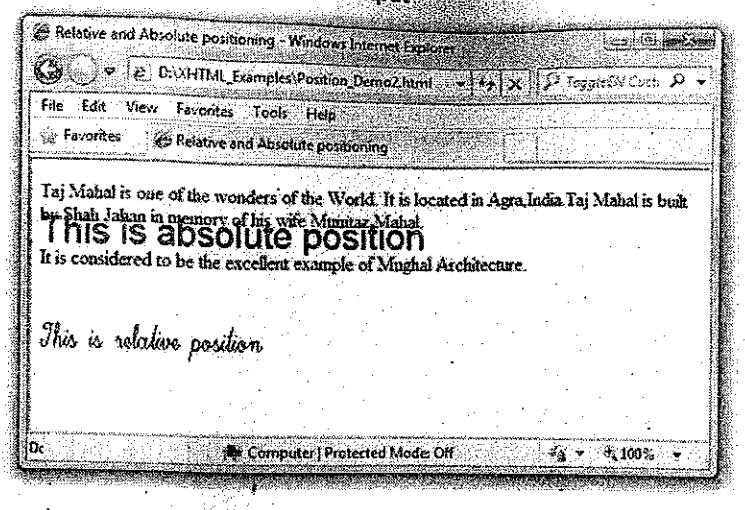
```
<!DOCTYPE html>
<html>
<head>
 <title>Relative and Absolute positioning</title>
 <style type="text/css">
 .text_pos1
 {
 position:relative;
 top:10px;
 font-family:script;
 font-size:30px;
 width:400px;
 }
 .text_pos2
 {
 position:absolute;
 top:10px;
 font-family:Arial;
 font-size:30px;
 width:400px;
 }

</style>
</head>
<body>
<p>Taj Mahal is one of the wonders of the World. It is located in Agra,India.Taj Mahal is
built by Shah Jahan in memory of his wife Mumtaz Mahal.
</p>
<p>It is considered to be the excellent example of Mughal Architecture.</p>
<p>This is relative position</p>
<p class="text_pos1">This is relative position</p>
<p class="text_pos2">This is absolute position</p>
</body></html>
```

Script Explanation

- In above script, we have defined two classes **text_pos1** and **text_pos2**. Both the classes have the same positional values i.e. left and top. Both are having same font-size and width parameter.
- Only font-family is changed to differ the look of both the elements. And the **text_pos1** has the position property set to relative position and **text_pos2** has the position property which is set to absolute position.
- Hence **text_pos1** (relative positioning) will be placed at the relative position from the previous text which is denoted by <p> tag.

- And absolute positioning will be the default positioning which does not consider the position of previous text.

Output



3.2.3 Static Positioning

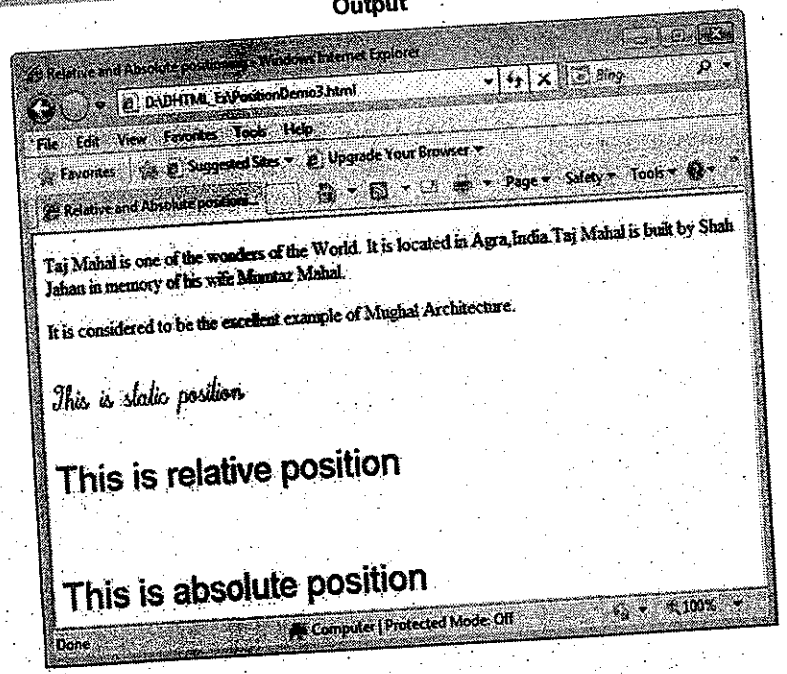
Definition : An element with **static** position always has the position the normal flow of the page gives it. This element just ignores the top,left,right and bottom declaration.

DHTML Document[PositionDemo3.html]

```
<!DOCTYPE html>
<html>
<head>
 <title>Relative and Absolute positioning</title>
 <style type="text/css">
 .text_pos1
 {
 position:static;
 font-family:script;
 font-size:30px;
 width:400px;
 }
 .text_pos2
 {
 position:relative;
 font-family:Arial;
 font-size:30px;
 width:400px;
```

```
}
text_pos3
{
position:absolute;
font-family:Arial;
font-size:30px;
width:400px;
}

</style>
</head>
<body>
<p>Taj Mahal is one of the wonders of the World. It is located in Agra,India.Taj Mahal is
built by Shah Jahan in memory of his wife Mumtaz Mahal.
</p>
<p>It is considered to be the excellent example of Mughal Architecture.</p>
<p class="text_pos1">This is static position</p>
<p class="text_pos2">This is relative position</p>
<p class="text_pos3">This is absolute position</p>
</body>
</html>
```

Output



Script explanation :

- In above script just add left and top parameter values in the class **text_pos3** and see the output on the browser. You may notice that there is no change in previous output and this one because static positioning neglects the left and top values.
- The statically displayed element cannot be displaced from its position.

Review Questions

1. *Explain different types of positioning with an example.* Question Bank, Nov.-11, Marks 10
2. *Explain absolute positioning and relative positioning.* Dec.-13, Marks 5

3.3 Moving Element

Dec.-13, Marks 5

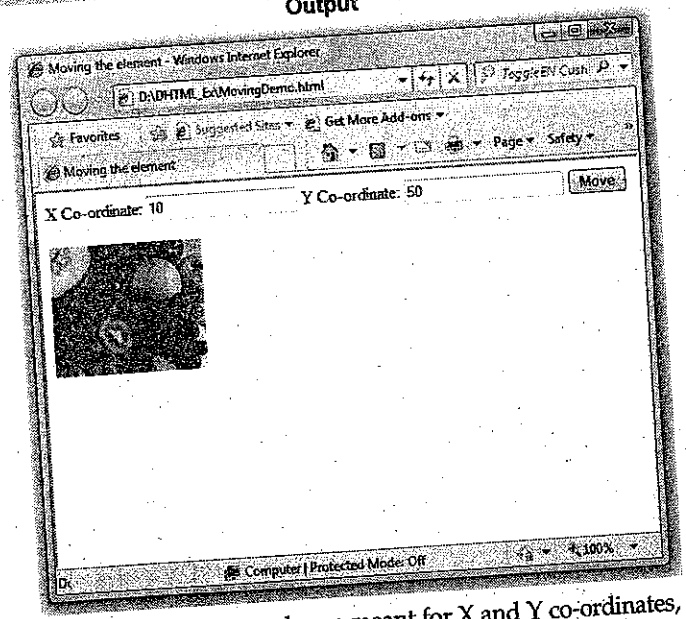
- The position attribute decide the position of an element on the web browser , we can change this position by simply changing its **left** and **top** values.
- Note that only **absolute** or **relative** elements can be moved because static elements just neglect the top and left values.
- In the following example, we have placed two input boxes and one button element on the form. With the help of these elements we want to move the image **"fruit.jpg"** which is placed on the web document.
- For moving this image we have invoked one function **"my_fun()"**. This function takes the values of X and Y co-ordinates through two text boxes that are supplied with and accordingly change the **left** and top values of the positioned element.
- The output shown along with this script itself is illustrative.

DHTML Document[MovingDemo.html]

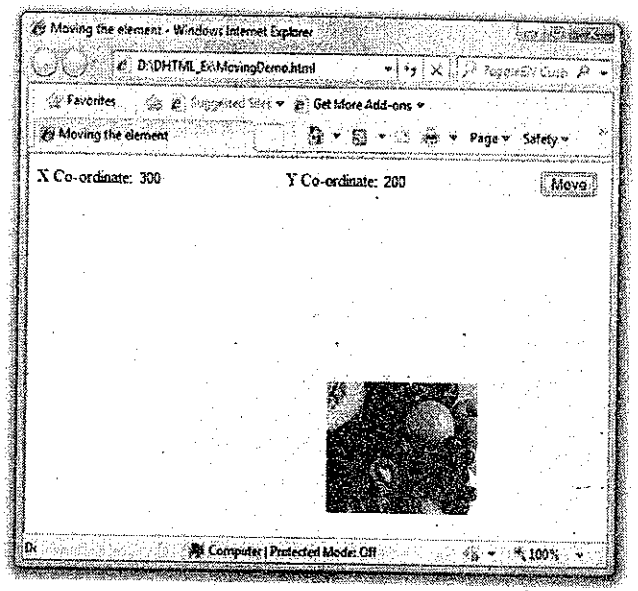
```
<!DOCTYPE html>
<html>
<head>
<title>Moving the element</title>
<script type="text/javascript">
function my_fun(x_pos,y_pos)
{
 var Dom_obj=document.getElementById("my_img").style;
 Dom_obj.top=y_pos+"px";
 Dom_obj.left=x_pos+"px";
}
</script>
```

```
</head>
<body>
 <form>
  <label>X Co-ordinate:
  <input type="text" value="10" id="x"/>
  </label>
  <label>Y Co-ordinate:
  <input type="text" value="50" id="y"/>
  </label>
  <input type="button" value="Move" onclick="my_fun(x.value,y.value);"/>
 </form>
 <div id="my_img" style="position:absolute;top:50px;left:10px;">
  <img src="fruit.jpg" />
 </div>
</body>
</html>
```

Output



Just change the values in two text boxes meant for X and Y co-ordinates, then click the Move button and you can see following sort of output.



Review Questions

1. *Illustrate Moving elements with simple example.* Question Bank, Marks 5
2. *Explain how to move an object from one place to another with an example* Dec. 13, Marks 5

3.4 Element Visibility

May-14, Marks 5

- We can have control over the visibility of particular element on the web document. This can be done using the property **visibility**.
- There are two values that can be set to the "visibility" and those are **visible** for getting the element displayed and **hidden** for hiding the element.
- Following is a simple DHTML document which makes use of this property.

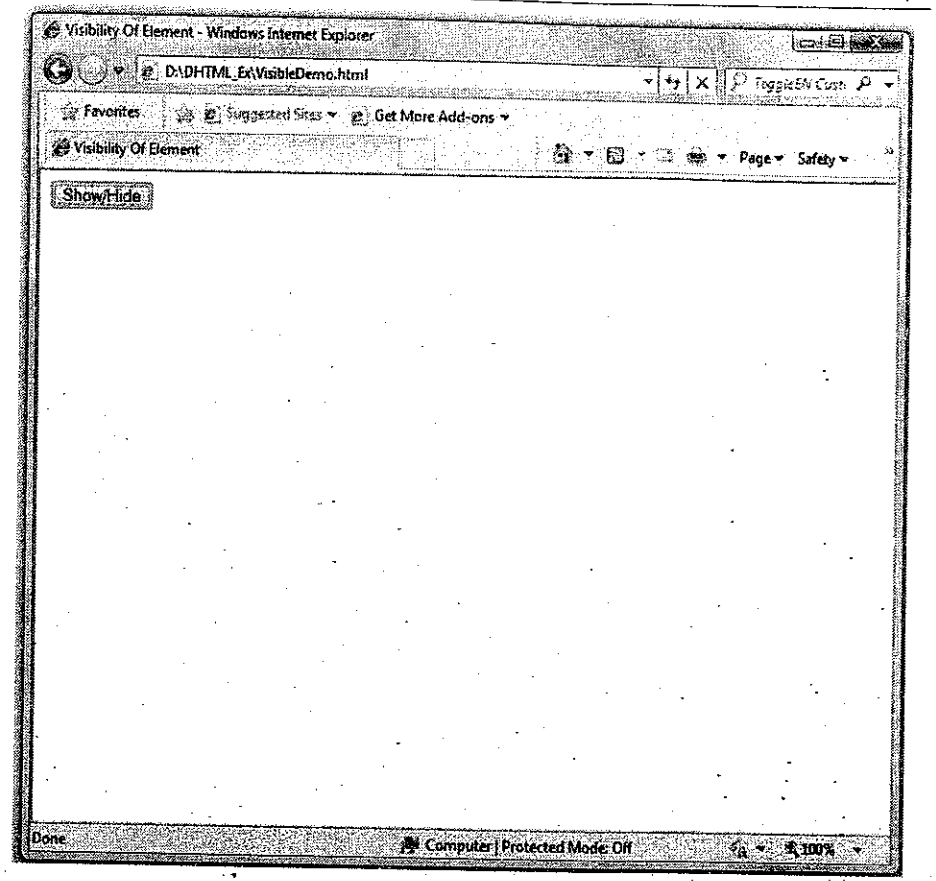
DHTML Document[VisibilityDemo.html]

```
<!DOCTYPE html>
<html>
<head>
<title>Visibility Of Element</title>
<script type="text/javascript">
function my_fun()
{
```

```
var Dom_obj=document.getElementById("my_img").style;
if(Dom_obj.visibility=="visible")
  Dom_obj.visibility="hidden";//Image will be as hidden
else
  Dom_obj.visibility="visible";//image will be displayed
}
</script>
</head>
<body>
<form>
<input type="button" value="Show/Hide" onclick="my_fun();"/>
</form>
<div id="my_img" style="position:absolute;top:10px;left:100px;">
<img src="bird.jpg" />
</div>
</body>
</html>
```

Output

Just click Show/Hide button repeatedly and get the effect of visibility demo.



Review Questions

1. *Explain Element visibility with simple example* — Question Bank, May-14, Marks 5
2. *Illustrate moving elements and element visibility with an example.* — Question Bank, Marks 10

3.5 Changing Colors and Fonts

Dec.-12, May-14, 15, Marks 10

3.5.1 Changing Colors

- There are two important properties – **backgroundColor** and **color** using which the background and foreground color can be changed.
- We can set the background color using **sytle.backgroundColor** attribute. For example

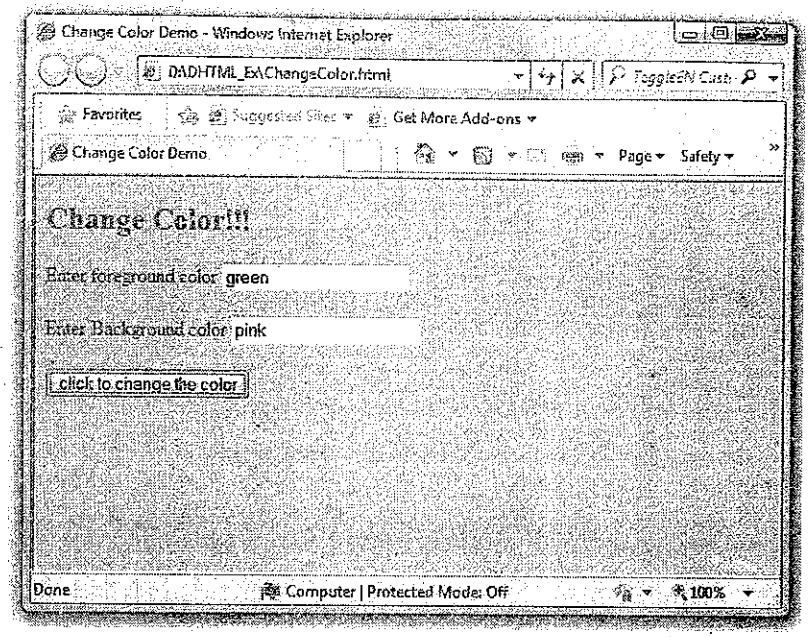
  document.body.style.backgroundColor = name_of_Color

- We can set the foreground color using **style.color** attribute. For example
  Document.body.style.color = name_of_color ;
- **Example Script :**

DHTML Document[ChangeColor.html]

```
<!DOCTYPE html>
<html>
<head>
<title>Change Color Demo</title>
<script type="text/javascript">
 function my_fun()
 {
  var F_color=document.getElementById("fore_color").value;
  var B_color=document.getElementById("back_color").value;
  document.body.style.backgroundColor=B_color;
  document.body.style.color=F_color;
 }
</script>
</head>
<body>
 <h2>Change Color!!!</h2>
 <form>
  <label>Enter foreground color
  <input type="text" value="" id="fore_color"/>
  </label>
  <br/><br/>
  <label>Enter Background color
  <input type="text" value="" id="back_color"/>
  </label>
  <br/><br/>
  <input type="button" value="click to change the color" onclick="my_fun();"/>
 </form>
</body>
</html>
```


Output



Review Question

1. *Explain with an example how to change dynamically background and foreground Colors of the document.*
   Question Bank, May 14, Marks 10

3.5.2 Changing Fonts

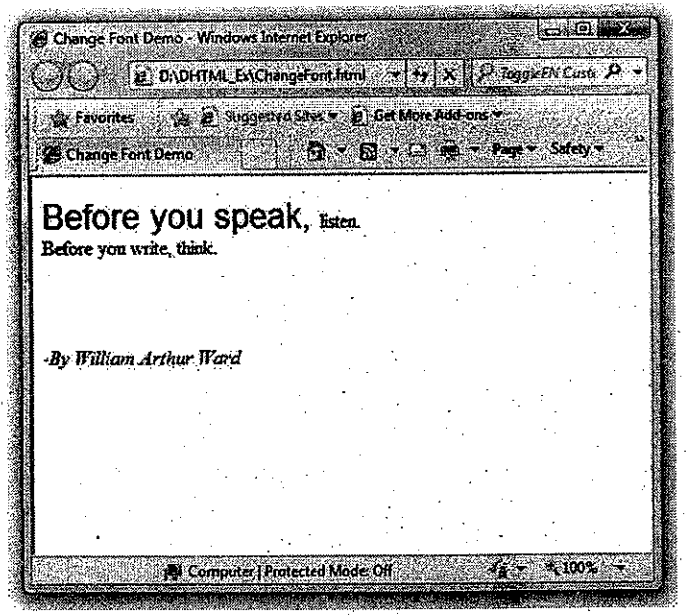
- Using the **style.font** the font style can be changed.
- For changing the font style some event must be triggered.
- In the following script we have used **onmouseover** and **onmouseout** events. On the **onmouseover** event we are making the text big, similarly we are changing its color and **onmouseout** event the text is brought to its normal style.
- **Example script :**

DHTML Document[ChangeFont.html]

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Change Font Demo</title>
</head>
<body>
```

```
<p>
 <a onmouseover="this.style.color='blue';
 this.style.font='20pt Arial';"
 onmouseout="this.style.color='black';
 this.style.font='12pt normal';">
  Before you speak,
 </a>listen.
 <br/>
 <a onmouseover="this.style.color='blue';
 this.style.font='20pt Arial';"
 onmouseout="this.style.color='black';
 this.style.font='12pt normal';">
  Before you write,
 </a>think.
 </p>
 <br/><br/>
 <p>
  <em>-By William Arthur Ward</em>
 </p>
</body>
</html>
```

Output



Review Question

1. *What event can be used to change a font when the mouse cursor is moved over and away from an element ? Illustrate with program.* Dec. 15, May 16, Marks 10

3.6 Dynamic Contents

The onblur Event

- Sometimes we need some helping messages when we make the entries in the text windows of the form.
- These all things can be achieved by setting the appropriate **onfocus** and **onblur** events.
- **Example script :**

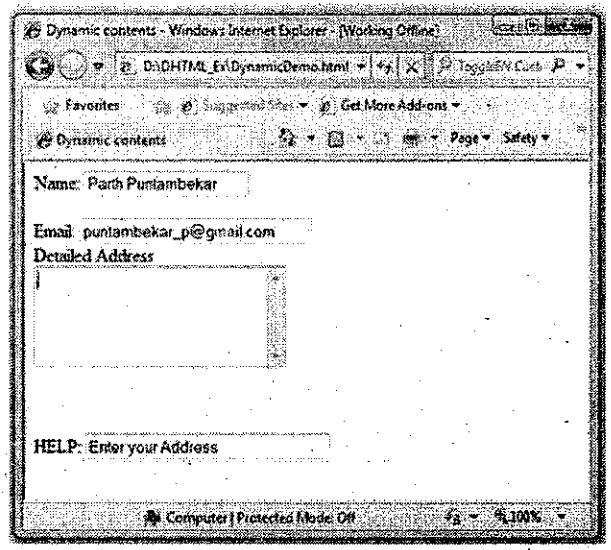
DHTML Script[DynameDemo.html]

```
<!DOCTYPE html>
<html>
<head>
<title>Dynamic contents</title>
<script type="text/javascript">
 var A =[ "Enter your name","Enter your email address","Enter your Address","Click on any
field to get the help" ];

 function My_fun(i)
 {
  form1.help_textbox.value=A[i];
 }
</script>
</head>

<body>
<form id = "form1" action = "">
Name: <input type = "text" size=20 name = "name"
onfocus = "My_fun(0)" onblur = "My_fun(3);"/><br/><br/>
Email: <input type = "text" size=30 name = "email"
onfocus = "My_fun(1)" onblur = "My_fun(3);"/><br/>
Detailed Address<br/>
<textarea name="txtarea1" rows = "5" cols="25"
onfocus= "My_fun(2)" onblur = "My_fun(3);"/>
</textarea> <br><br><br><br>
HELP: <input type = "text" size=32 name = "help_textbox" />
</form>
</body>
</html>
```

Script Explanation

- In the above script, we have created two **textboxes** for storing the user name and e-mail ID.
- One text area is created for storing the address. We have created one more textfield in which the helping messages will be displayed.
- If we click on the **'name'** text box then that means focus is on the 'name' text box. Hence the helping message 'Enter your name' will be displayed in the help text box.
- The **onblur** event works just opposite to the **onfocus** event. Here is the sample output.

Output



The change Event

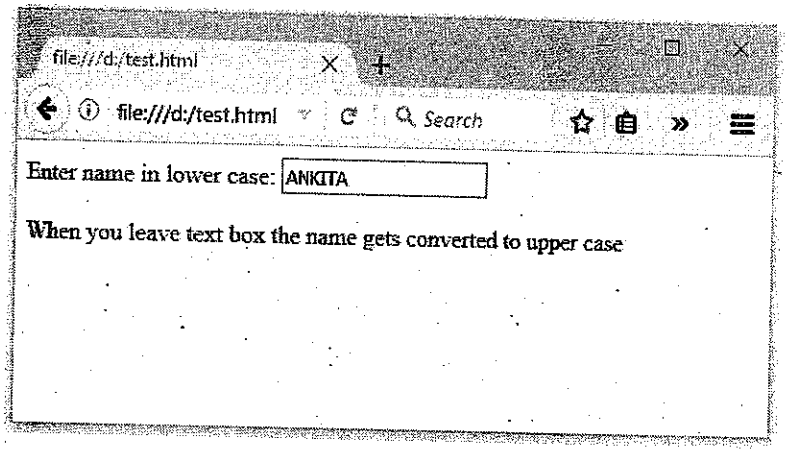
- The **onchange** event occurs when the value of an element has been changed.
- The function that makes necessary changes in the value of the elements is invoked on **onchange** event.
- **Example Script :**

```
<!DOCTYPE html>
<html>
<body>
<script>
function changetoUpper()
{
```



```
  var obj = document.getElementById("txt_id");
  obj.value = obj.value.toUpperCase();
}
</script>
Enter name in lower case: <input type="text" id="txt_id"
onchange="changetoUpper()">
<p> When you leave text box the name gets converted to upper case</p>
</body>
</html>
```

Output



Script Explanation : In above script

- We have created an input text box which accepts the name in lower case.
- **The onchange** event is called for which the event handler function executes.
- In this function the contents in the text box are converted and displayed in upper case
- This text box has an id **txt_id**. This id is passed to the **getElementById** function and the value written in this textbox can be grabbed.
- Using **toUpperCase()** function the text can be changed to upper case.

Review Question

1. *Illustrate how to handle blur event and change event with an example.* **Question Bank, Marks 10**

3.7 Stacking Elements

May-12, Dec.-16, Marks 10

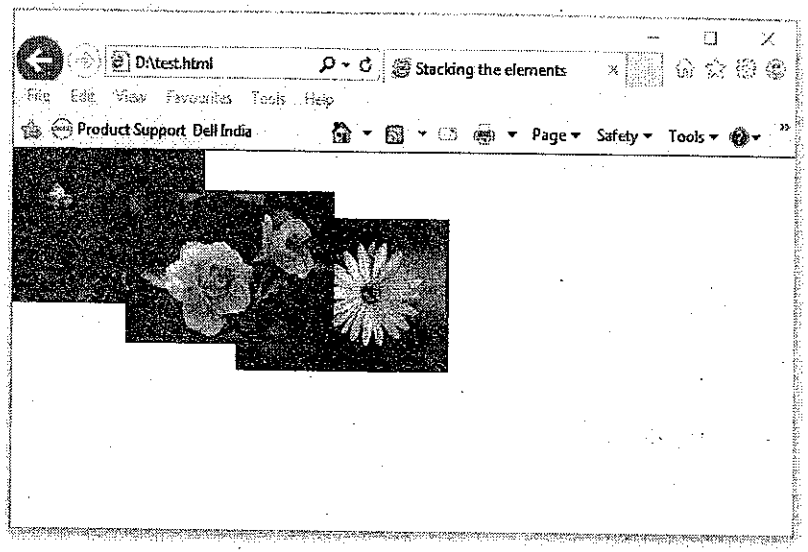
- Using DHTML we can have overlapping elements and clicking the particular element from this overlapped stack that element appears front.

- This effect can be achieved by a special attribute called z-index. This effect can be achieved by controlling the z-co-ordinate of the element. That means the element whose z-index is greater than all other elements will be displayed over the other elements.
- In the following example, we have placed three passages of the same object using the absolute position. And on mouse click event one function is called in which the z-index of corresponding element is increased.
- **Example Script :**

DHTML Document[Test.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Stacking the elements</title>
  <script type="text/javascript">
    var top="first";
    function my_fun(id) {
    var Dom_obj=document.getElementById(top).style;
    var Dom_new=document.getElementById(id).style;
    Dom_obj.zIndex="0";
    Dom_new.zIndex="30";
    top=id;
    }
  </script>
  <style type="text/css">
    .img1 {
    position:absolute;top:0;left:0;z-index:0;border:2px solid;background-color:pink;
    }
    .img2 {
    position:absolute;top:30px;left:90px;z-index:0;border:2pxsolid;background-color:pink;
    }
    .img3 {
    position:absolute;top:50px;left:180px;z-index:0;border:2px solid;background-color:pink;
    }
  </style>
</head>
<body>
<img class="img1" id="first" src="img1.jpg" onclick="my_fun(id);"/>
<img class="img2" id="second" src="img2.jpg" onclick="my_fun(id);"/>
<img class="img3" id="third" src="img3.jpg" onclick="my_fun(id);"/>
</body>
</html>
```


Output



Review Question

1. *Illustrate with an example for dynamic stacking of images*

Question Bank,Model Question Paper, May-12, Dec-16, Marks 10

3.8 Locating the Mouse Cursor

May-13, 15, 16, Marks 10

We can locate the position of the mouse with the help some mouse event. The mouse click event is implemented by **MouseEvent** interface. The mouse position can be denoted by two co-ordinates i.e. x and y co-ordinates.

There are two ways of specifying the position of the mouse.

- **Co-ordinates with respect to browser window**

Using the clientX and clientY the x and y positions of mouse event can be obtained. These positions are calculated from the upper left corner of browser window.

- **Co-ordinates with respect to computer screen**

Using the screenX and screenY the x and y positions of mouse event can be obtained. These positions are calculated related to clients computer screen.

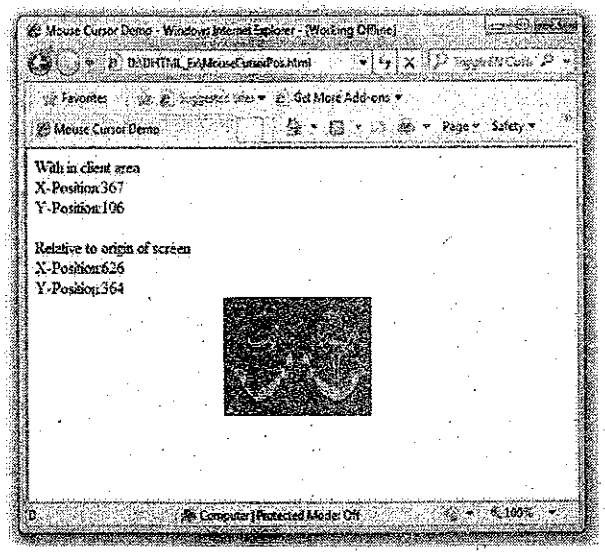
But normally used of clientX and clientY is preferred for obtaining the x and y positions when some mouse event occurs on the browser window.

In the following document we are displaying both the types of co-ordinates.

DHTML Document[MouseCursorPos.html]

```
<!DOCTYPE html>
<html>
<head>
 <title>Mouse Cursor Demo</title>
 <script type="text/javascript">
 function my_fun(e)
 {
  points1.innerText="With in client area"+"\n"+"X-Position:"+e.clientX+"\n"+
"Y-Position:"+e.clientY;
  points2.innerText="\n"+"Relative to origin of screen"+"\n"+"X-Position:"
+e.screenX+"\n"+"Y-Position:"+e.screenY;
 }
 </script>
</head>
<body onclick="my_fun(event);">
 <span id="points1">(0,0)</span>
 <br/>
 <span id="points2">(0,0)</span>
 <div align="center">
  <img src="lamp1.jpg"/>
 </div>
</body>
</html>
```

Output



Script Explanation :

- In above script we have used **innerText** attribute.
- If certain block of text gets displayed and may get changed repeatedly then the innerText property is used.
- Onclick event we are passing the object event to the event handler function.
- Hence we can obtain the current positions.

Review Question

1. *Explain how to locate the mouse cursor.* **Question Bank, May 13, 15, 16, Marks 10**

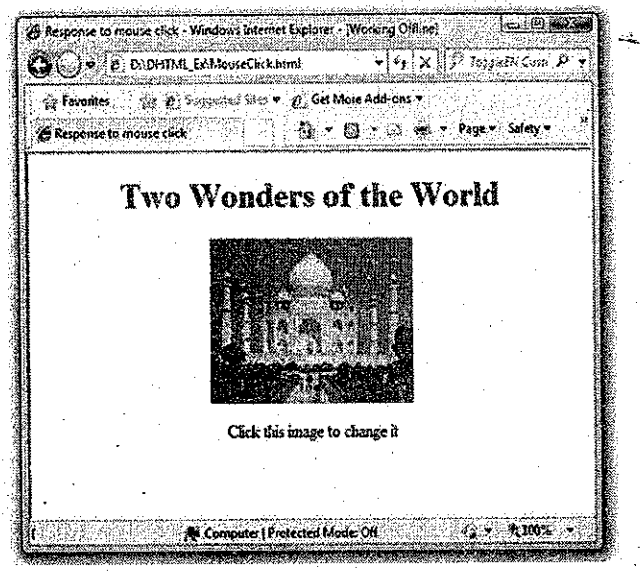
Reacting to a Mouse Click

- When we click the mouse button it goes down and then comes back. These activities can be caught using the **onmousedown** and **onmouseup** events respectively.
- Following is an example in which we will display one image when user presses mouse button and another image gets displayed when user releases the mouse button.

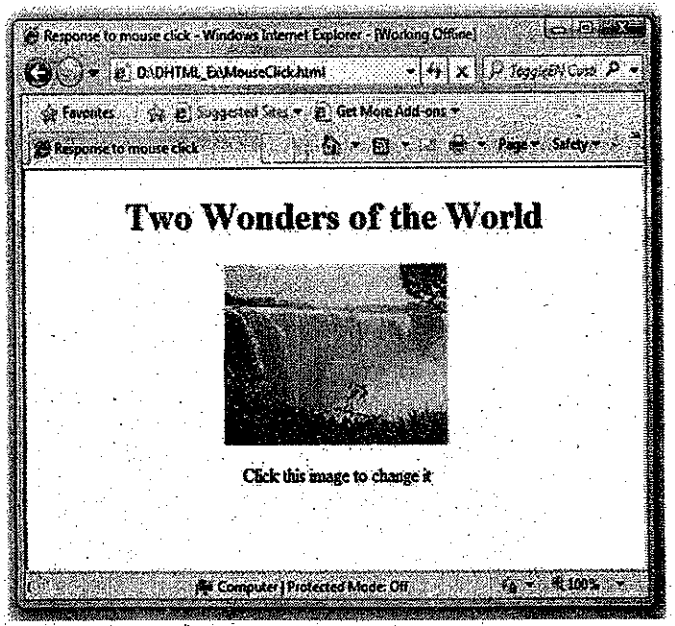
DHTML Document[MouseClick.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Response to mouse click</title>
<script type="text/javascript">
function my_fun()
{
 my_img.src="waterfall.jpg";
}
function my_fun1()
{
 my_img.src="TajMahal.jpg";
}
</script>
</head>
<body>
 <center>
  <h1> Two Wonders of the World</h1>
  <img id="my_img" onmousedown="my_fun()" onmouseup="my_fun1()"
src="TajMahal.jpg" width="200" height="150" />
  <p> Click this image to change it</p>
 </center>
</body>
</html>
```

Output



If click on the image we will get another image being displayed.



3.10 Slow Movement of the Elements

Nov. 11, May 12, Marks 5

- The element can be moved by changing the **left** and **top** properties.
- For moving an element slowly we have to move it by small amount repetitively.
- Each move must be separated by some amount of time.
- There are two methods defined by an object **window** which are responsible for moving an element. These methods are **setTimeout** and **setInterval.**
- The **setTimeout** function takes two parameters.
  1. The first one is the function which is responsible for moving an element.
  2. And the second parameter is the delay time given in milliseconds. The function which is responsible for moving an element can be called with the specified amount of delay. For example

Function Delay

↓ ↓

```
setTimeout("Move1();",10)
```

means the function **Move1()** can be called after 10 milliseconds repeatedly.

- **setTimeout** : It executes a function after waiting a specified number of miliseconds.

  Syntax : setTimeout (function, milliseconds)

- **setInterval :** It is same as setTimeout ( ) but function executes continuously.

  Syntax : setInterval (function, milliseconds)

- Both of these functions are used for moving the elements. Both are same. The only difference between the two is that **setInterval** fires again and again in intervals while **setTimeout** fires once.
- Following is a simple DHTML document in which the text is moved in forward and backward direction using **setTimeout** function.

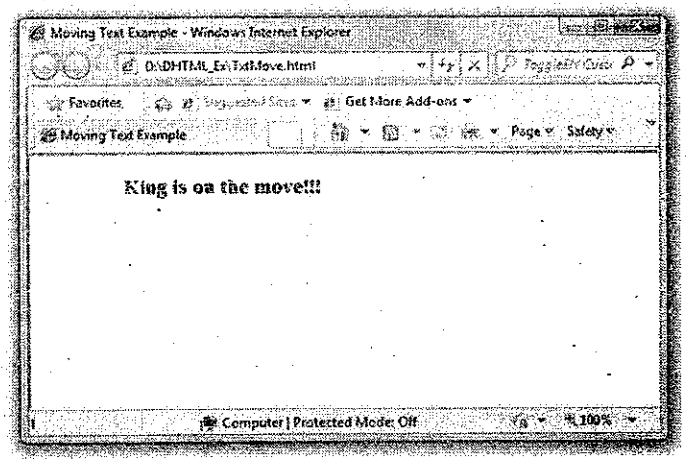
DHTML Document[TxtMove.html]

```
<!DOCTYPE html>
<html>
<head>
<title>Moving Text Example</title>
<script language="javascript">
var x=0;
var direction=1;
function Move1()
{
  Dom obj=document.getElementById("txt");
```

```
  x += dir;
  if (x >= 300 || x <= -300) dir = 0 - dir;
  Dom_obj.style.left=x;
  window.setTimeout("Move1();",10);
 }
</script>
</head>
<body onload="Move1();">
<h3 align="center" style="position: relative;" ID="txt">
King is on the move!!!
</h3>
</body>
</html>
```

Moves horizontally. If we change to DOM_obj.style.top=x then the text will move in vertical direction.

Output



Script explanation :

- In above script, we have called a **Move1()** function in which we are accessing the <h3> elements whose id is "txt".
- Then an object **Dom_obj** is created. The value of x will be incremented by one each time and is assigned to the left property of the object **Dom_obj**.
- Then **setTimeout** function will be called. This method calls the function **Move1()** again with the time delay of 10 milliseconds.
- Thus on each call of **Move1()** function the value of x will be incremented and is assigned to the **left** property of **Dom_obj**. This continues until the value being reached is 300.
- On reaching the value to 300, the value of x gets decremented. This causes this to move the element in backward direction.
- Hence as an output we get the text is moving forward and backward.

Review Questions

1. *Describe the parameters and actions of the setTimeout and setInterval functions.*
   Question Bank, Model Question Paper, May 12, Marks 5
2. *Explain the use of SetTimeOut() function with its syntax* **Nov. 11, Marks 5**

3.11 Dragging and Dropping the Elements

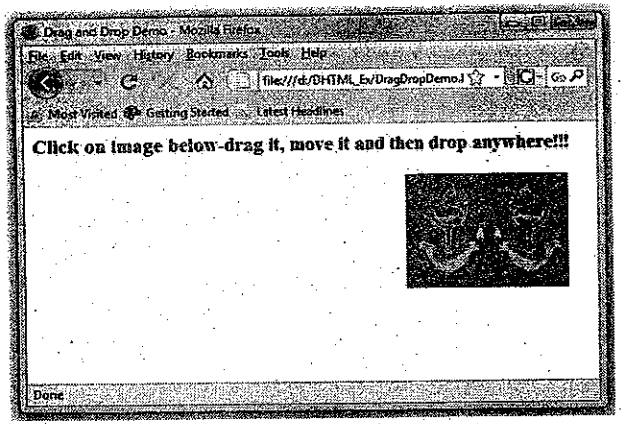
- Using the **mouseup**, **mousedown** and **mousemove** is used for dragging and dropping the element on the browser window.
- In the following script we have written three basic functions for that purpose.
- We have declared three global variables because they will then be available to all the three handlers.
- Using the **currentTarget** property we can get the element to be dragged and dropped. When user presses the mouse button down the **Catch_function** is called. And when user moves the mouse or releases the mouse button up then **Move_function** and **Drop_function** will be invoked. This invoking is done using **DOM 2 event model.**
- While moving the element we make use of **left** and **top** properties. The **clientX** and **clientY** properties give the current x and y co-ordinates.

DHTML Document[DragDropDemo.html]

```
<html>
<head>
 <title> Drag and Drop Demo</title>
 <script type = "text/javascript">
  var X_offset, Y_offset, Item;
  function Catch_function(e)
  {
    Item = e.currentTarget;
    var posX = parseInt(Item.style.left);
    var posY = parseInt(Item.style.top);
    X_offset = e.clientX - posX;
    Y_offset = e.clientY - posY;
    document.addEventListener("mousemove", Move_function, true);
    document.addEventListener("mouseup", Drop_function, true);
    e.stopPropagation();
    e.preventDefault();

  }
  function Move_function(e)
  {
```

```
    Item.style.left = (e.clientX - X_offset) + "px";
    Item.style.top = (e.clientY - Y_offset) + "px";
    e.stopPropagation();
  }
  function Drop_function(e)
  {
    document.removeEventListener("mouseup", Drop_function, true);
    document.removeEventListener("mousemove", Move_function, true);
    e.stopPropagation();
  }
  </script>
</head>
<body>
  <h3>
  Click on image below-drag it, move it and then drop anywhere!!!
    <br /><br />
    <img src="lamp1.jpg"
    style = "position: absolute;
    top: 100px;
    left: 100px;
    border:2px solid;"
    onmousedown = "Catch_function(event);" />
  </h3>
</body>
</html>
```

Output



Just experiment the above script by dragging the image anywhere on the browser window and dropping it. Note that the output of this document can be seen in Firefox web browser because Internet Explorer does not support it.

4 Introduction to XML

Syllabus

Introduction, The Syntax of XML, XML Document Structure, Document Type Definitions, Declaring Elements, Declaring Attributes, Declaring Entities, A Sample DTD, Internal and External DTDs, Namespaces, XML Schema, Schemas Fundamentals, Defining the Schema, Defining the Schema Instances, An Overview of Data types, Simple Types, Complex Types, Displaying Raw XML Documents, Displaying XML Documents with CSS.

Contents

4.1 Introduction

XML stands for eXtensible Markup Language.

This scripting language is similar to HTML. That means, this scripting language contains various tags. But these tags are not predefined tags, in-fact user can define his own tags.

Thus HTML is designed for representation of data on the web page whereas the XML is designed for transport or to store data.

4.2 Syntax of XML

Nov.11, May 16, 12, Mark 5

- In XML we can create our own tags. In fact XML allows the programmer to create markup for virtually any type of information.
- In XML the basic entity is **element**. The elements are used for defining the tags. The elements typically consist of opening and closing tag. Mostly only one element is used to define a single tag.
- The syntax of writing any element for opening tag is ***<element name>***
- The syntax of writing any closing element for closing tag is ***</element name>***
- For example -

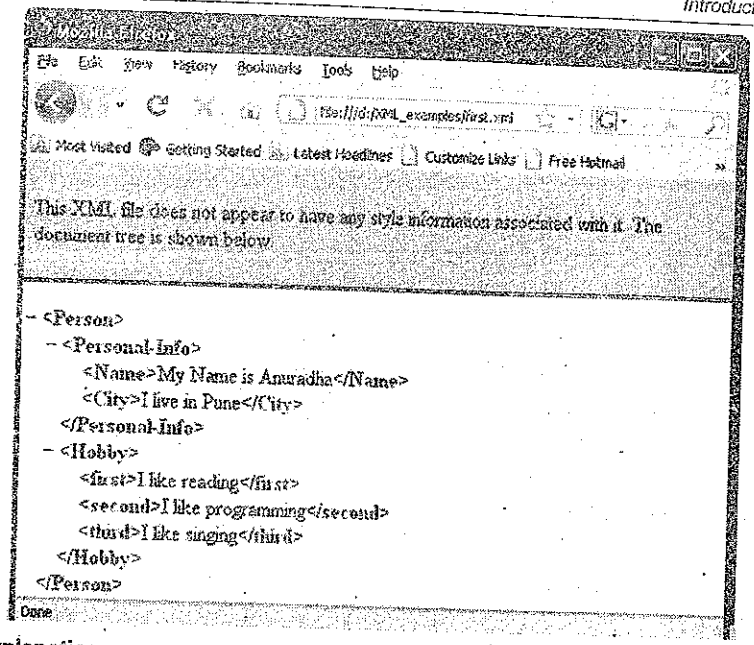
```
<Student> I am learning Computer Engineering </Student>
```

- Every XML must have a single root element. **For example -** Consider following XML document, in which <person> is a root element.
- **For example :** Here is our first XML program which can be written using notepad or using some standard XML editor.

XML Document [name: first.xml]

```
<Person>
   <Personal-Info>
      <Name>My Name is Anuradha</Name>
      <City>I live in Pune</City>
   </Personal-Info>
   <Hobby>
      <first>I like reading</first>
      <second>I like programming</second>
      <third>I like singing</third>
   </Hobby>
</Person>
```

- Now open some web browser like Internet Explorer or Mozilla Firefox and just observe the **output** of above program as follows -



Script explanation :

The file name extension used for xml program is **.xml.** Hence the name of above program is *first.xml*. The XML scripts are **self descriptive.** As you can see that the information being displayed on the web page is about a person whose name is *Anuradha*. This person lives in city *Pune* and her hobbies are *reading*, *programming* and *singing*.

- **Rules that must be followed while writing XML -**

Here are some rules that must be followed while writing the XML programs -

1. XML is a case sensitive. For example -

```
<hobby>I like drawing</Hobby> or
<Hobby>I like drawing</hobby>
```

is not allowed because the words **hobby** and **Hobby** are treated differently.

2. In XML each start tag must have matching end tag. For example -

```
<staring> XML is funny to write
<simple> it is simple to implement </simple>
```

That means a closing tag is a must.

3. The elements in XML must be properly nested. For example -

```
<one><two>Hello how are you?</one></two>  is wrong but
<one><two>Hello how are you?</two></one>  is correct
```

4.2.1 Features of XML

Following are some features which are most suitable for creating web related applications.

- XML is EXtesible Markup Language intended for transport or storage of the data.
- The most important feature of XML is that user is able to define and use his own tag. This allows us to restrict the use of the set of tags defined by proprietary vendors. On the other hand other markup languages like HTML or SGML requires a predefined set of tags.
- XML contains only data and does not contain any formatting information. Hence document developers can decide how to display the data.
- XML permits the document writer to create the tags for any type of information. Due to this virtually any kind of information can be such as mathematical formulae, chemical structures or some other kind of data can be described using XML.
- Searching sorting, rendering or manipulating the XML document is possible using extended stylesheet language (XSL).
- The XML document can be validated using some external tools.
- Some commonly used web browsers like Internet Explorer and Firefox Mozilla provide support to the tags in XML. Hence XML is not at all vendor specific or browser specific.

4.2.2 Difference between XML and HTML

Sr. No.	HTML	XML
1.	HTML stands for Hypertext Markup Language.	XML stands for eXtensible Markup Language.
2.	HTML is designed to display data with the focus on look and feel of data.	XML is used to transport and store data, with focus on what data is.
3.	HTML is case insensitive.	XML is case sensitive.
4.	HTML has predefined tags.	XML has custom tags can be defined and the tags are invented by the author of the XML document.
5.	As HTML is for displaying the data it is static.	As XML is for carrying the data it is dynamic.
6.	It can not preserve white space.	It can preserve the white space.

Review Questions

1. *Write a note on XML.*
2. *List the features of XML.* Question Bank, Nov. 11, Marks 5
3. *Difference between XML and HTML.* May 12, Marks 5
4. *List XML syntax rules.* May 16, Marks 5

4.3 XML Document Structure

Nov. 11, May 14

- An XML document consists of one or more entities that are logically related collection of information. The complete XML document is called as **document entity.**
- The complete XML document is broken into multiple entities.
- It is a good practice to break the large document into smaller part, so that the entire document becomes manageable.
- The XML document can not store certain information such as images or graphics. This type of information is generally stored as a binary data. Usually this binary data is stored in a separate file and is referred as **binary entities.**
- There are some characters in XML which have special meanings. For example if we insert the "<" or ">" character within the text then it will generate an error because these characters are used for enclosing the tags. Hence we use

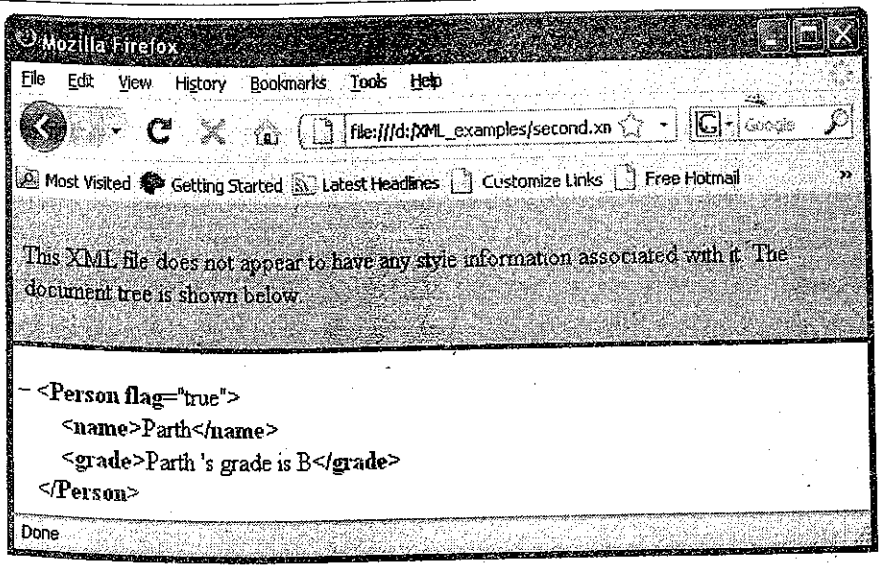
<	for less than
>	for greater than
&apos;	for apostrophe
&amp;	for ampersand
&quot;	for double quotes

- For example -

XML Document

```
<Person flag="true">
    <name>Parth</name>
    <grade>Parth &apos;s grade is B</grade>
</Person>
```

The **output** will be



4.3.1 Advantages of XML

1. XML document is human readable and we can edit any XML document in simple text editors.
2. The XML document is language neutral. That means a Java program can generate an XML document and this document can be parsed by Perl.
3. Every XML document has a tree structure. Hence complex data can be arranged systematically and can be understood in simple manner.
4. XML files are independent of an operating system.

4.3.2 Uses of XML

1. XML is used to display the meta contents i.e. XML describes the content of the document.
2. XML is useful in exchanging data between the applications.
3. The data can be extracted from database and can be used in more than one application. Different applications can perform different tasks on this data.

Review Question

1. *Write a note on XML document structure.* **Question Bank - Nov.-11, May-14, Marks 5**

4.4 Document Type Definition

May-12, 16, Dec.-14, Marks 5

- The document type definition is used to define the basic building block of any xml document.
- Using DTD we can specify the various **elements types, attributes** and their **relationship** with one another. Basically DTD is used to **specify the set of rules** for structuring data in any XML file.
- For example : If we want to put some information about some students in XML file then, generally we use tag **student** followed by his/her **name, address, standard and marks.** That means we are actually specifying the manner by which the information should be arranged in the XML file. And for this purpose the Document Type Definition is used.
- Various building blocks or **keywords** of XML are -

1. Elements

The basic entity is **element.** The elements are used for defining the tags. The elements typically consist of opening and closing tag. Mostly only one element is used to define a single tag.

2. Attribute

The attributes are generally used to specify the values of the element. These are specified within the double quotes.

For example -

```
<flag type="True">
```

The type attribute of the element flag is having the value True.

3. CDATA

CDATA stands for character data. This character data will be parsed by the parser.

4. PCDATA

It stands for Parsed Character Data (i.e. text).

4.4.1 Declaring Elements

- The element type data can be declared using the ELEMENT.
- The syntax for defining the ELEMENT is

```
<!ELEMENT elt_name (elt1,elt2,...)>
```

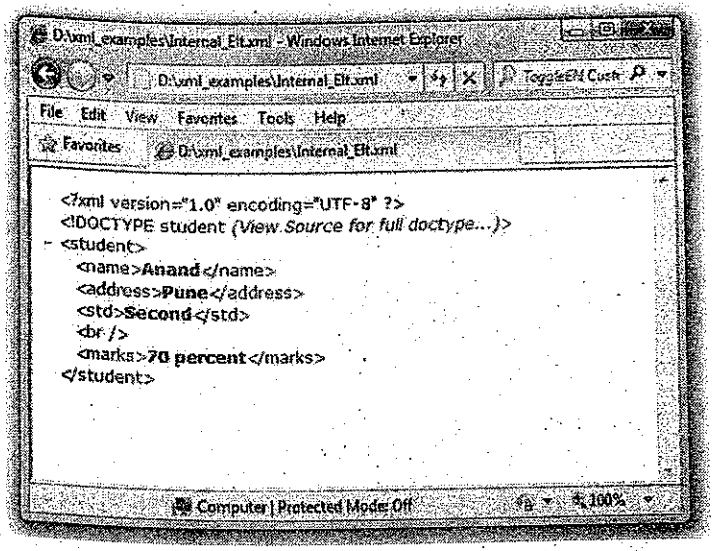
- **Example**

XML Document[Internal_Elt.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE student [
<!ELEMENT student (name,address,std,br,marks)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT address (#PCDATA)>
<!ELEMENT std (#PCDATA)>
<!ELEMENT br EMPTY>
<!ELEMENT marks (#PCDATA)>

]>
<student>
   <name>Anand</name>
   <address>Pune</address>
   <std>Second</std>
   <br />
   <marks>70 percent</marks>
</student>
```

Here we are defining the DTD internally

Output



- The ELEMENT must be defined in the beginning of the XML file. It must be enclosed within **<!DOCTYPE *root_element* ....>**

- There are various ways by which the actual element can be **defined**. It is enlisted in the following table -

Category	Syntax	Description
Empty	<!ELEMENT *elt_name* EMPTY>	Empty elements can be declared.
Character-Data	<!ELEMENT *elt_name* (#PCDATA)>	Character data can be declared.
Any content	<!ELEMENT *elt_name* ANY>	Any combination of data can be declared.
Sequence	<!ELEMENT *elt_name* (elt1,elt2,...)>	Sequence of elements that must appear in specific order.

- Sometimes we can define the ELEMENT along with the **iterator characters**. The iterator characters are the special character which are associated with some meaning.
- Some **commonly used iterator characters** are as given below -

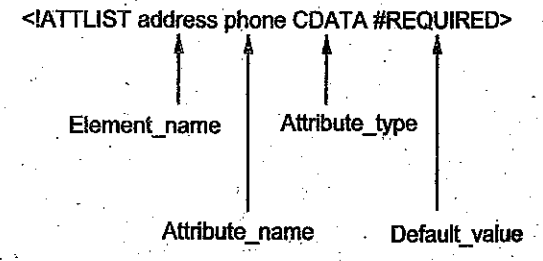
Iterator character	Meaning	Example
?	The preceding choice appears zero or once.	<!ELEMENT (*marks?*)> The child element marks can appear for zero or one time.
*	The preceding choice appears for any number of time(zero or more times).	<!ELEMENT (*person**)> The person can appear for any number of times.
+	The preceding choice appears for at least once (one or more times).	<!ELEMENT (*person+*)> The person can appear for at least once.

4.4.2 Declaring Attributes

- The attribute type declaration is done using the string **ATTLIST.**
- The attribute type specifies the type of data which can be used for specifying the attribute. The syntax for specifying the attribute declaration is as follows -

```
<!ATTLIST element_name attribute_name attribute_type default_value>
```

- **For example**



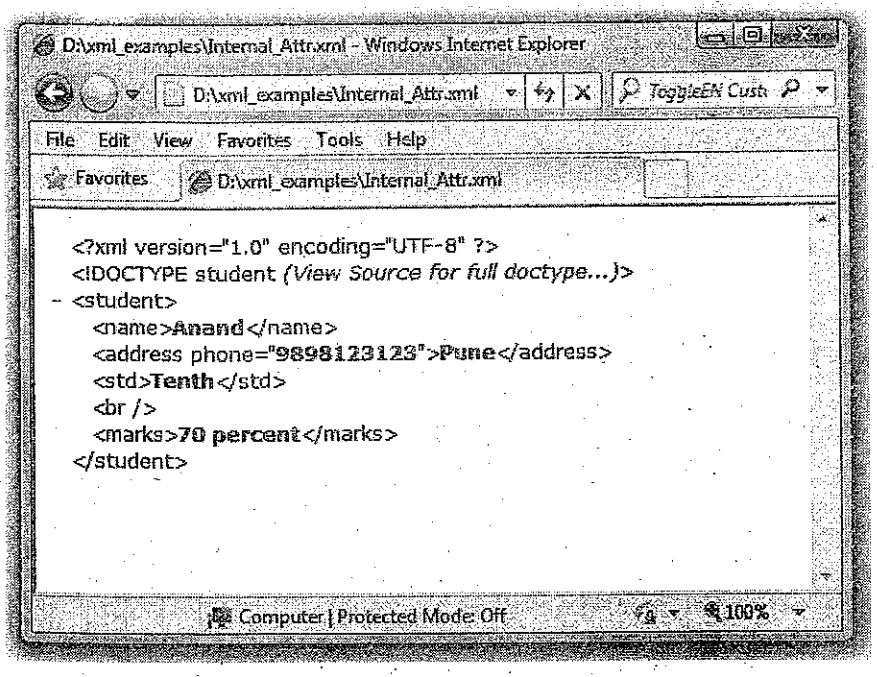
- The XML document containing attributes is as given below -

XML Document[Internal_Attr.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE student [
<!ELEMENT student (name,address,std,marks)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT address (#PCDATA)>
<!ELEMENT std (#PCDATA)>
<!ELEMENT marks (#PCDATA)>

<!ATTLIST address phone CDATA #REQUIRED>

]>
<student>
    <name>Anand</name>
    <address phone="9898123123">Pune</address>
    <std>Tenth</std>
    <br />
    <marks>70 percent</marks>
</student>
```

Using **ATTLIST** the attribute can be defined.

Output



- Following are some attribute type -

Attribute Type	Syntax	Description
character data	CDATA	Attribute value is character data.
Enumerated	(str1|str2|...)	Attribute value must be one from an enumerated list.
Identifier	ID	Attribute value is a unique id.
Identifier Reference	IDREF	Attribute value is the id of another element.
Identifier Reference list	IDREFS	Attribute value is a list of other ids.
Name Token	NMTOKEN	Attribute value is a valid XML name.
Name Token list	NMTOKENS	Attribute value is a list of valid XML names.

The default_values are -

Default Value	Meaning
value	Default value can be defined. For example - <!ATTLIST Index flag="0"> In XML document <Index flag="1" /> If the flag is not assigned with any value then by default it will be 0.
#REQUIRED	Required attribute forces the attribute to be present even if there is no default value.
#IMPLIED	The default value is not provided and attribute can be optionally used.
#FIXED>value	The attribute value is fixed and cannot be changed.

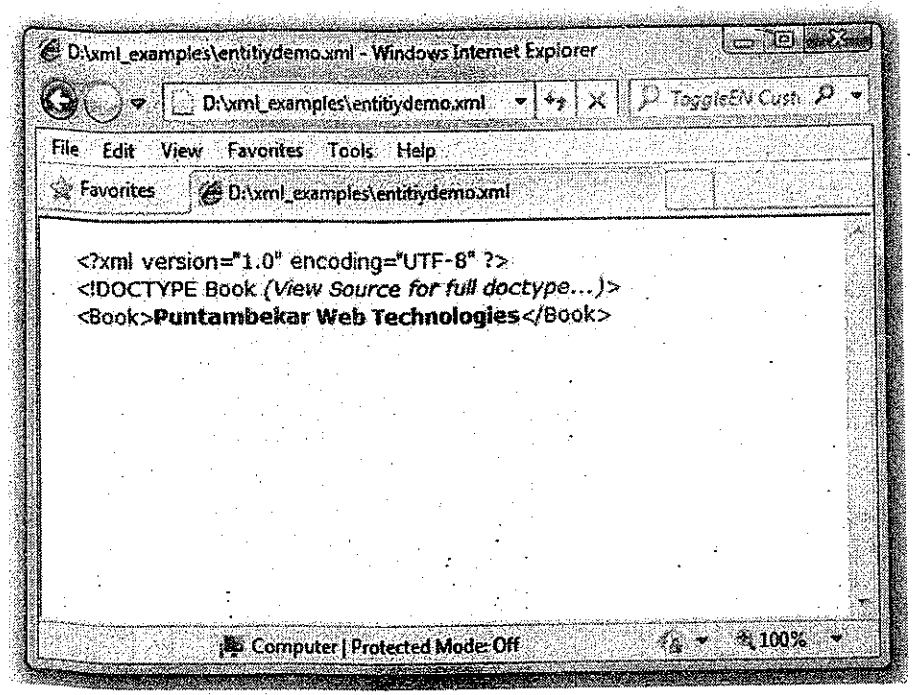
4.4.3 Declaring Entities

- The attribute type declaration is done using the string **ENTITY.**
- Inside the XML file at the beginning the entity can be defined using the <!DOCTYPE ...> and with the help of string ENTITY. Following XML document demonstrates it -

XML Document [EntityDemo.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Book[
<!ENTITY writer "Puntambekar">
<!ENTITY nbsp " ">
<!ENTITY title "Web Technologies">
]>
<Book>&writer; &title;</Book>
```

Output



Note that we can define the predefined entities ** ** along with the other entity declaration.

The entity can be used using following syntax

```
&name_of_entity;
```

In above script we have used two entities namely **writer** and **title**. For displaying the space within these two entities we have used an entity **nbsp** and for defining it we have used **#160.**

4.4.4 Internal and External DTDs

- There are two types of DTDs and those are **Internal DTD** and **External DTD.**
- Following examples illustrate these types -

1. Internal DTD

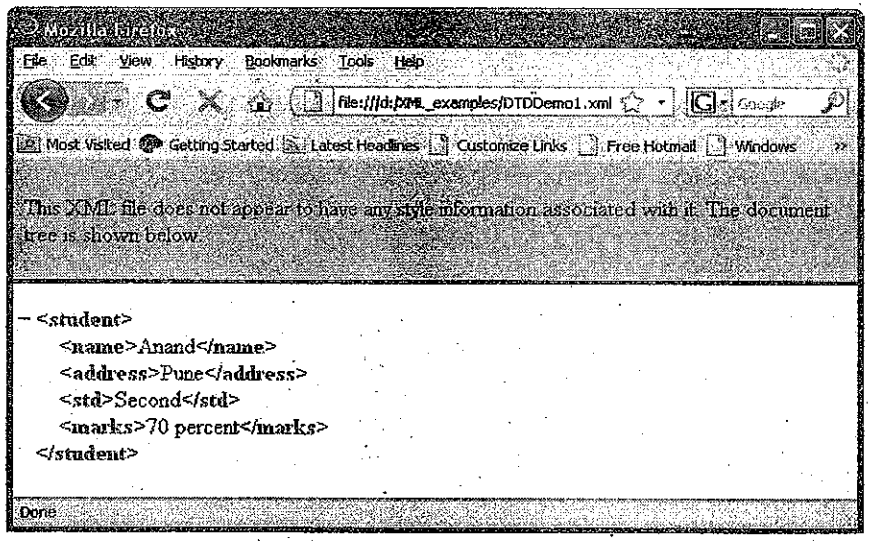
Consider following xml document -

XML Document *[DTDDemo1.xml]*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE student [
<!ELEMENT student (name,address,std,marks)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT address (#PCDATA)>
<!ELEMENT std (#PCDATA)>
<!ELEMENT marks (#PCDATA)>
]>
<student>
    <name>Anand</name>
    <address>Pune</address>
    <std>Second</std>
    <marks>70 percent</marks>
</student>
```

Here we are defining the DTD internally

Output



2. External DTD

In this type, an external DTD file is created and its name must be specified in the corresponding XML file. Following XML document illustrates the use of external DTD.

Step 1 : Creation of DTD file *[student.dtd]*

Open some suitable text editor or a notepad. Type following code into it -

```
<!ELEMENT student (name,address,std,marks)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT address (#PCDATA)>
<!ELEMENT std (#PCDATA)>
<!ELEMENT marks (#PCDATA)>
```

Now save this file as *student.dtd*

Step 2 : Creation of XML document

XML Document *[DTDDemo.xml]*

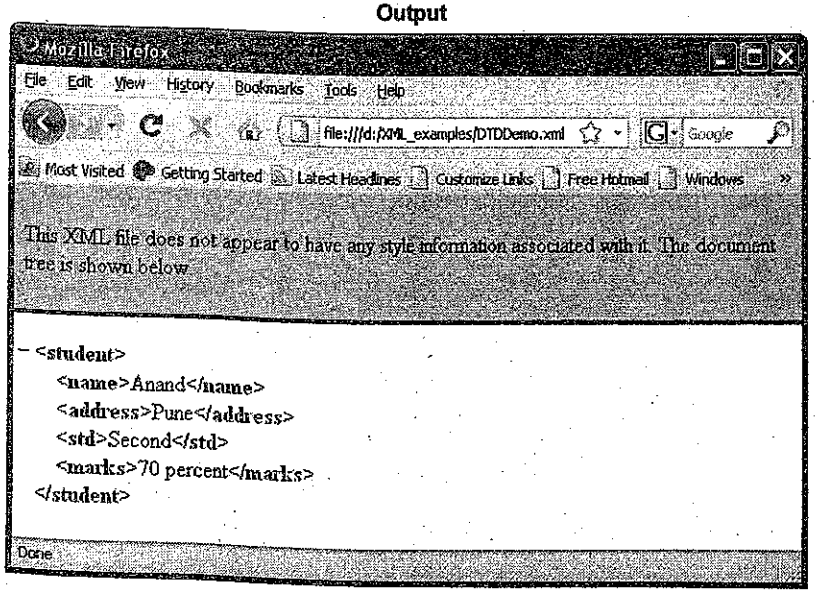
Now create a XML document as follows -

```
<?xml version="1.0"?>
<!DOCTYPE student SYSTEM "student.dtd">
<student>
   <name>Anand</name>
   <address>Pune</address>
   <std>Second</std>
   <marks>70 percent</marks>
</student>
```

The external DTD file is created.

Step 3 : Using some web browser open the XML document.

Output



Merits of DTD

1. DTDs are used to define the **structural components** of XML document.
2. These are relatively **simple and compact.**
3. DTDs can be defined **inline** and hence can be embedded directly in the XML document.

Demerits of DTD

1. The DTDs are very basic and hence cannot be much specific for complex documents.
2. The language that DTD uses is not an XML document. Hence various frameworks used by XML cannot be supported by the DTDs.
3. The DTD cannot define the type of data contained within the XML document. Hence using DTD we cannot specify whether the element is numeric or string or of date type.
4. There are some XML processor which do not understand DTDs.
5. The DTDs are not aware of namespace concept.

Review Questions

1. *Define DTD. Mention four possible keywords in DTD declaration.* Question Bank, Model Question Paper, May-12, Marks 5
2. *Explain internal and external DTD's with an example.* Question Bank, Marks 5
3. *Illustrate declaring of elements, attributes and entities in DTD with an example.* Question Bank, Model Question Paper, Marks 10
4. *What are the four possible parts of an attribute declaration in a DTD ?* Dec.-14, Marks 5
5. *What is DTD ? Explain different types of DTD.* May-16, Marks 5

4.5 Namespaces

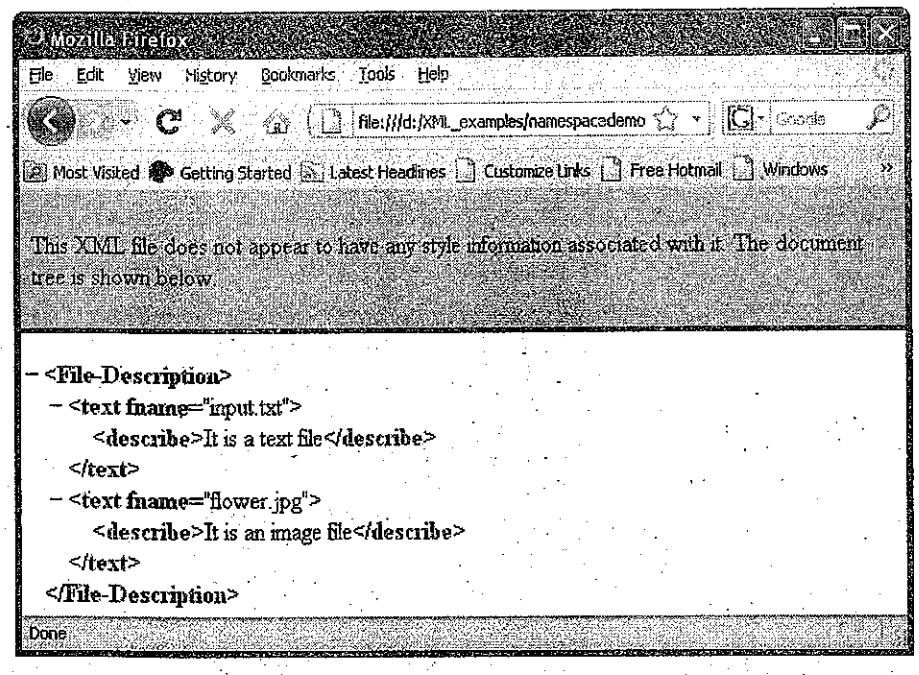
Dec.-12, May-13, 14, 16, Marks 5

- Sometimes we need to create two different elements by the same name.
- The XML document allows us to create **different elements** which are having the **common name**. This technique is known as **namespace.**
- In some web documents it becomes necessary to have the same name for two different elements. Here different elements mean the elements which are intended for different purposes. In such a case support for namespace technique is very much helpful.
- For example : Consider the following XML document.

XML Document [name:namespacedemo.xml]

```
<File-Description>
   <text fname="input.txt">
      <describe>It is a text file</describe>
   </text>
   <text fname="flower.jpg">
      <describe>It is an image file</describe>
   </text>
</File-Description>
```

The above document does not produce any error although the element text is used for two different attribute values. The output will be



Review Questions

1. *Explain how to declare namespace with example.* — Question Bank, Marks 10
2. *What is XML namespace ? Explain with example.* — Dec. 12, May-13, 14, 16, Marks 5

4.6 XML Schema

Nov. 11, May-13, 14, 15, Dec. 16, Marks 10

4.6.1 Schema Fundamentals

- The XML schemas are used to represent the structure of XML document.
- These can be used as an alternative to XML DTD. The XML schema language is called as **XML Schema Definition (XSD) language**.
- The XML schema became the **World Wide Web Consortium (W3C) recommendation in 2001.**
- XML schema defines **elements, attributes, elements having child elements, order of child elements.** It also defines fixed and default values of elements and attributes.
- XML schema also allows the developer to use **data types.**

Advantages of schema

1. The schemas are more specific.
2. The schema provide the support for data types .
3. The schema is aware of namespaces.
4. The XML schema is written in XML itself and has a large number of built in and derived types.
5. The XML schema is the W3C recommendation. Hence it is supported by various XML validator and XML processors.

Disadvantages of schema

1. The XML schema is complex to design and hard to learn.
2. The XML document cannot be if the corresponding schema file is absent.
3. Maintaining the schema for large and complex operations sometimes slows down the processing of XML document.

Advantages of schema over DTD

1. Both the schemas and DTDs are useful for defining structural components of XML. But the DTDs are basic and cannot be much specific for complex operations. On the other hand schemas are more specific.
2. The schemas provide support for defining the type of data. The DTDs do not have this ability. Hence content definition is possible using schema.
3. The schemas are namespace aware and DTDs are not.

4. The XML schema is written in XML itself and has a large number of built in and derived types.
5. The schema is the W3C recommendation. Hence it is supported by various XML validator and XML processors but there are some XML processors which do not support DTD.
6. Large number of web applications can be built using XML schema. On the other hand relatively simple and compact operations can be built using DTD.

4.6.2 Defining the Schema

- The scheme is defined in xsd file in which the desired structure of the XML document is defined.

  For example -
- This file contains the complex type with four child simple type elements.

XML Schema [StudentSchema.xsd]

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="Student">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="address" type="xs:string"/>
            <xs:element name="std" type="xs:string"/>
            <xs:element name="marks" type="xs:string"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

Script explanation :

- The xs is qualifier used to identify the schema elements and types.
- The document element of schema is **xs:schema**. The xs:schema is the root element. It takes the attribute **xmlns:xs** which has the value **http://www.w3.org/2001/XMLSchema**. This declaration indicates that document should follow the rules of XML schema. The XML schema rules are defined by the **W3 recommendation in year 2001.**
- Then comes **xs:element** which is used to define the **xml element.**
- In above case the element **Student** is of complex type who have four child elements : *name, address, std* and *marks*. All these elements are of simple type **string**.

4.6.3 Defining the Schema Instances

- After defining the xml schema we have to develop XML document in which the desired values to the XML elements can be given.
- Following is a schema instance which calls the **StudentSchema.xsd** at the beginning.

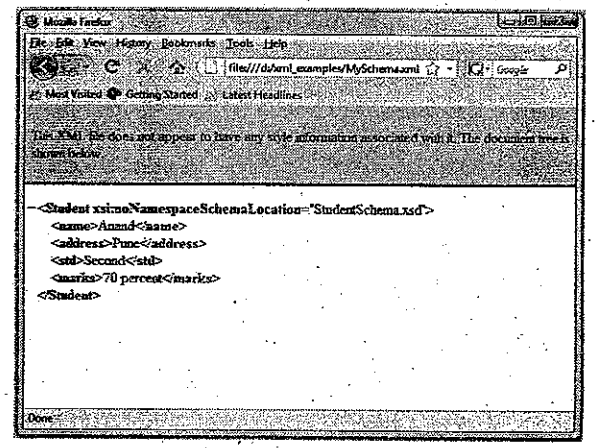
XML Document [*MySchema.xml*]

```
<?xml version="1.0" encoding="UTF-8"?>
<Student xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="StudentSchema.xsd">
<name>Anand</name>
<address>Pune</address>
<std>Second</std>
<marks>70 percent</marks>
</Student>
```

Script explanation

- In above XML document, the first line is typical in which the version and encoding is specified.
- There is **xmlns:xsi** attribute of document which indicates that XML document is an instance of XML schema. And it has come from the namespace **"http://www.w3.org/2001/XMLSchema-instance**
- To tie this XML document with the some schema definition we use the attribute **xsi:noNamespaceSchemaLocation** .The value that can be passed to this attribute is the name of xsd file **"StudentSchema.xsd".**
- After this we can define the child elements.

Output



4.6.4 An Overview of Data Types

Various data types are that can be used to specify the data types of an element are -

- String
- Date
- Numeric
- Boolean

Let us discuss each of these data types with the help of examples -

1. String Data Type

The string data type is used to define the element containing characters, lines, tabs or white spaces. Following schema definition illustrates this data type.

Step 1 : First we will write a schema definition in which the data type of element Student_Name is declared as of string type.

XML Schema [stringType.xsd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="Student_Name" type="xs:string" />
</xs:schema>
```

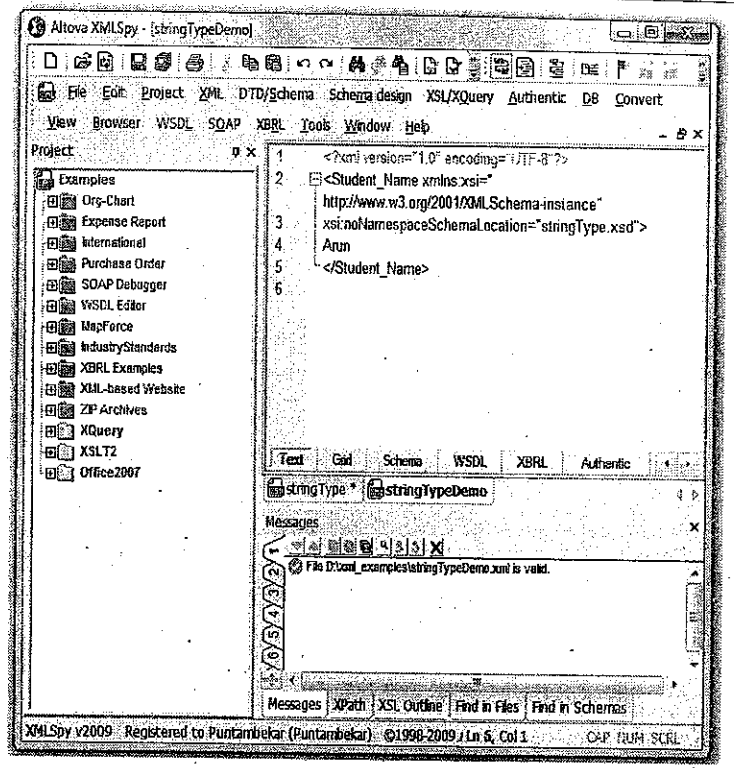
Step 2 : Then we must write the corresponding XML document in which the above given schema definition is used.

XML Document [stringTypeDemo.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<Student_Name xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="stringType.xsd">
Arun
</Student_Name>
```

In above XML document the element **Student_Name** is there for which we have written the name "Arun". That means string type of data is assigned to the element **Student_Name.**

Step 3 : Then we can validate our XML document. The following screenshot shows that the above created XML document is valid.



2. Date Data Type

For specifying the date we use date data type. The format of this date is yyyy-mm-dd where yyyy denotes the year, mm denotes the month and dd specifies the day. While specifying the date if any entry is single digit then it must be with leading zero. All the entries(i.e. yyyy,mm and dd) must be present.

Here is an illustration.

XML Schema [dateType.dtd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="Date_of_Birth" type="xs:date"/>
</xs:schema>
```

XML Document [dateTypeDemo.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<Date_of_Birth xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="dateType.xsd">
2009-10-01
</Date_of_Birth>
```

```
</Date_of_Birth>
```

We can also specify the time by specifying the time as follows

XML Document

```
<?xml version="1.0" encoding="UTF-8"?>
<Date_of_Birth xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="dateType.xsd">
2009-09-01-01:57
</Date_of_Birth>
```

Suppose we want to specify the time only then

XML Schema [timeType.xsd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="MyTime" type="xs:time"/>
</xs:schema>
```

XML Document [timeTypeDemo.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<MyTime xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="timeType.xsd">
01:57:03
</MyTime>
```

Time must be specified in the form of hh:mm:ss Where
hh means hour
mm means minute
ss means seconds

3. Numeric

If we want to use numeric value for some element then we can use the data types as either **decimal** or **integer**.

XML Schema [decimalDemo.xsd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="My_Marks" type="xs:decimal"/>
</xs:schema>
```

XML Document [decimalTypeDemo.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<My_Marks xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="decimalDemo.xsd">
99.99
</My_Marks>
```


Using decimal data type we can specify the value that may contain some decimal point. We can have another data type called integer which is used to specify the numeric values that are without any decimal point.

XML Schema [integerDemo.xsd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="My_Number" type="xs:integer"/>
</xs:schema>
```

XML Document [integerTypeDemo.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<My_Number xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="decimalDemo.xsd">
1      <---------- specifying the numeric value without any decimal point
</My_Number>
```

4. boolean

For specifying the **true** or **false** values we must us the boolean data type.

XML Schema [booleantype.xsd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="Flag" type="xs:boolean"/>
</xs:schema>
```

XML Document [booleantypeDemo.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<Flag xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="booleanType.xsd">
false
</Flag>
```

4.6.5 Simple Types

- XML defines the simple type which contains only text and does not contain any other element or attributes. But the text that appears for element comes along with some type.
- The syntax of using the simple types in XML schema is as follows -

```
xs:element name = "Element_name" type = "data_type"
```

- Here **type** can be any built in data type -

```
xs:string
xs:boolean
xs:decimal
xs:date
xs:time
```

- For example

```
<name>Anand</name>
<age>38</age>
```

Can be written in XML schema as

```
<xs:element name = "name" type= "xs:string" />
<xs:element name= "age" type="xs:integer" />
```

- We can also specify the **fixed** or **default** values to the **simple elements**. For example - When we want to set the default value to the element **then we** use the word **default** when no other value is specified to the element then the **default** value gets assigned.

```
<xs:element name="Mango" type = "xs:string" default="sweet"/>
```

- For setting the fixed value we use the word **fixed**. After specifying the fixed value we can not specify any other value to the element.

```
<xs:element name="Mango" type = "xs:string" fixed="sweet"/>
```

- We can specify the simple user defined data type using **the simpleType** element. But for that we need to put some restrictions on the contents.

These restrictions are called as **fecets**.

- For example : The Xml schema definition file can be written as follows :

Month.xsd

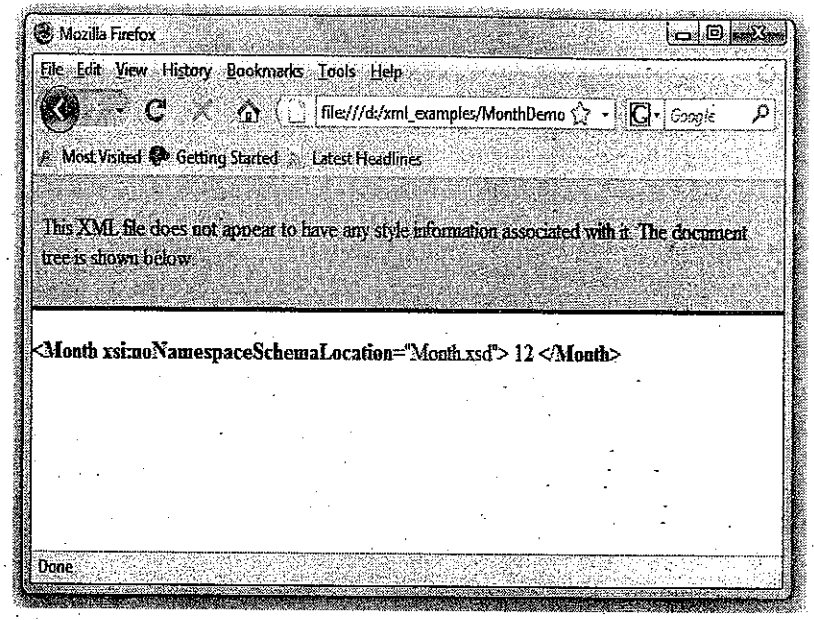
```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="Month">
        <xs:simpleType>
            <xs:restriction base="xs:integer">
                <xs:minInclusive value="1"/>
                <xs:maxInclusive value="12"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
</xs:schema>
```


- Now we will write corresponding XML document as follows -

XML Document [MonthDemo.xml]

```
<?xml version="1.0"?>
<Month xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Month.xsd">
12
</Month>
```

output



4.6.6 Complex Types

- Complex elements are the elements that contain some other elements or attributes.
- There are several types of complex type elements. In this section we will discuss the complex type elements **that contain elements only.**
- All the complex types are defined with **complexType** tag.
- The elements that are contained in the complex type must be in ordered group, unordered group, a choice or a named group. This can be done with the help of indicators.
- **Indicators** allow us to define the elements in the manner we want.
- Following are some indicators -

  1. all 2. choice 3. sequence 4. maxoccurs 5. minoccurs

4.6.6.1 The All Indicator

For this type of indicator, we can specify that all the child elements must occur in any sequence but each child must appear at least once.

XML Schema [complexType5.xsd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <!--This schema definition is for indicator all -->
   <xs:element name="Student">
      <xs:complexType>
         <xs:all>
            <xs:element name="Stud_Name" type="xs:string"/>
            <xs:element name="Roll_No" type="xs:integer"/>
         </xs:all>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

XML Document [complexType5Demo.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<Student xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="complexType5.xsd">
<Stud_Name>Jayashree</Stud_Name>
<Roll_No>10</Roll_No>
<!-- If either Stud_Name or Roll_No is missing then this xml doument will be invalid.
Both the elements should be present because it has indicator all -->

</Student>
```

Note that in above document all the child elements i.e. **Stud_Name** and **Roll_No** must be present. If any one of them is missing then it will make the XML invalid.

4.6.6.2 The Choice Indicator

Using the choice indicator we can select the element of our own choice. In that case it is not necessary to have all the elements present in the XML document. Here is an illustration.

XML Schema [complexType6.xsd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <!--This schema definition contains choice indicator-->
   <xs:simpleType name="Stud_Name">
      <xs:restriction base="xs:string"/>
   </xs:simpleType>
```



```
   <xs:simpleType name="Roll_No">
      <xs:restriction base="xs:integer"/>
   </xs:simpleType>
   <xs:element name="Student">
      <xs:complexType>
         <xs:choice>
            <xs:element name="Stud_Name" type="Stud_Name"/>
            <xs:element name="Roll_No" type="Roll_No"/>
         </xs:choice>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

Note that we have used the indicator **xs:choice**. There are two types defined "Stud_Name" and "Roll_No". For defining these two types we have used **SimpleType**.

Then the XML document containing "Student" element is defined as follows -

XML Document [complexType6Demo.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<Student xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="complexType6.xsd">
<Stud_Name>Jayashree</Stud_Name>
<!--It takes either Stud_Name or Roll_No at a time but not the both-->
</Student>
```

Note that in above XML document if we use both the child elements i.e. Stud_Name and Roll_No at a time then it will generate an error.

4.6.6.3 The Sequence Indicator

For specifying that the child elements must be present in some specific order, **sequence** indicator can be used.

XML Schema [complexType7.xsd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="Student">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="Stud_Name" type="xs:string"/>
            <xs:element name="Roll_No" type="xs:integer"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

XML Document [ComplexTypeDemo7.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<Student xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="complexType7.xsd">
<Stud_Name>Jayashree</Stud_Name>
<Roll_No>10</Roll_No>
<!--It takes both Stud_Name and Roll_No but in a given sequence-->
<!--If Roll_No is taken before Stud_Name then this xml document will be invalid-->
</Student>
```

Note that in above XML document the child elements Stud_Name and Roll_No must appear in the same sequence as is given XML schema (i.e. complexType7.xsd). If we change the sequence of these child elements in the XML file we will get an error.

4.6.6.4 The maxOccurs Indicator

This indicator is used to denote the maximum number of times an element can occur. Following XML schema makes use of maxOccurs indicator.

XML Schema [complexType8.xsd]

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="Student">
      <xs:complexType>
      <xs:sequence>
         <xs:element name="Stud_Name" type="xs:string"/>
         <xs:element name="Roll_No" type="xs:integer" maxOccurs="3"/>
      </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

XML Document *[complexType8Demo.xml]*

```
<?xml version="1.0" encoding="UTF-8"?>
<Student xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="complexType8.xsd">
<Stud_Name>Jayashree</Stud_Name>
<Roll_No>10</Roll_No>
<Roll_No>20</Roll_No>
<Roll_No>30</Roll_No>
<!--It takes both Stud_Name and Roll_No but in a given sequence-->
<!--Maximum occurrence of element Roll_No should be less than or equal to 3-->
</Student>
```

In above XML document, the element Roll_No should appear maximum 3 times. But if Roll_No appears for more than 3 times then an error occurs.

4.6.6.5 The minOccurs Indicator

This indicator is used to denote the minimum number of times an element can occur. Following XML schema makes use of minOccurs indicator.

XML Schema *[complexType9.xsd]*

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="Student">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="Stud_Name" type="xs:string"/>
            <xs:element name="Roll_No" type="xs:integer"
               minOccurs="1" maxOccurs="10"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

Review Questions

1. *Differentiate between simple type and complex type XML elements.* Question Bank, Marks 5
2. *Explain different XSD indicators.* Question Bank, Model Question Paper, Nov.-11, May-14, Marks 10
3. *Define XML schema. List the advantage of XML schema over DTD.* Question Bank, Nov.-11, May-13, Marks 5
4. *Explain schema and schema instance with an example.* Question Bank, May-14, Marks 10
5. *Explain how to create simple type and complex type element with an example.* Question Bank, May-13, 15, Marks 10
6. *Explain with example, how to create complex type elements.* Dec.-16, Marks 10

4.7 Displaying Raw XML Documents

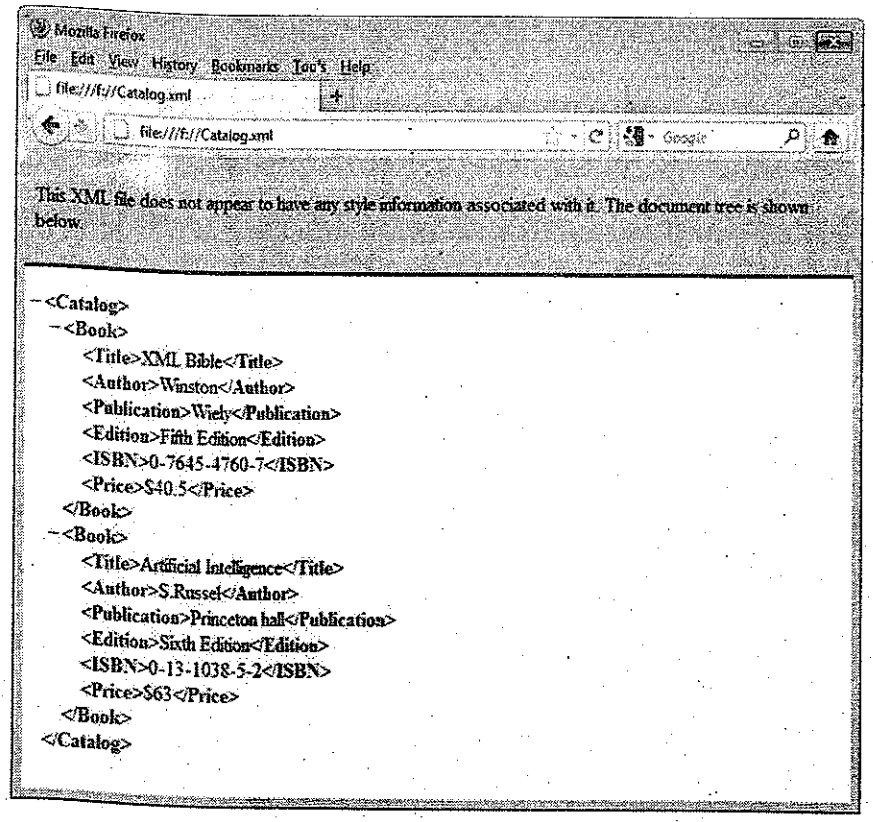
- The web browser basically does not know how to format tags defined in web document.
- If we display the XML document without the style-sheet then it is displayed in a listing manner.
- For example :

XML Document[Catalog.xml]

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

```
    <Title>XML Bible</Title>
    <Author>Winston</Author>
    <Publication>Wiely</Publication>
    <Edition>Fifth Edition</Edition>
    <ISBN>0-7645-4760-7</ISBN>
    <Price>$40.5</Price>
  </Book>
  <Book>
    <Title>Artificial Intelligence</Title>
    <Author>S.Russel</Author>
    <Publication>Princeton hall</Publication>
    <Edition>Sixth Edition</Edition>
    <ISBN>0-13-1038-5-2</ISBN>
    <Price>$63</Price>
  </Book>
</Catalog>
```

The output will be as follows -



4.8 Displaying XML Documents with CSS

May 12, Dec. 12, Marks 10

As we have seen many XML documents on browser look just similar to those written on the notepad. This is because browser does not know anything about the formatting or about the user defined tags. But if we want to apply some style to these documents we can use **Cascading Style Sheet(CSS)**. Following XML document illustrates the use of CSS in XML document.

Step 1 :

We will first write a simple XML document in which the library book details are given

XML Document [name:CSSDemo.xml]

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/css" href="library.css"?>
<Catalog>
  <Book>
    <Title>XML Bible</Title>
    <Author>Winston</Author>
    <Publication>Wiely</Publication>
    <Edition>Fifth Edition</Edition>
    <ISBN>0-7645-4760-7</ISBN>
    <Price>$40.5</Price>
  </Book>
  <Book>
    <Title>Artificial Intelligence</Title>
    <Author>S.Russel</Author>
    <Publication>Princeton hall</Publication>
    <Edition>Sixth Edition</Edition>
    <ISBN>0-13-1038-5-2</ISBN>
    <Price>$63</Price>
  </Book>
  <Book>
    <Title>Java 2</Title>
    <Author>Watson</Author>
    <Publication>BPB Publications</Publication>
    <Edition>Third Edition</Edition>
    <ISBN>0-41-1058-7-2</ISBN>
    <Price>$63</Price>
  </Book>
```

The first line denotes the version and encoding for xml document

The second line says that this document has a stylesheet. The name of stylesheet is given using href

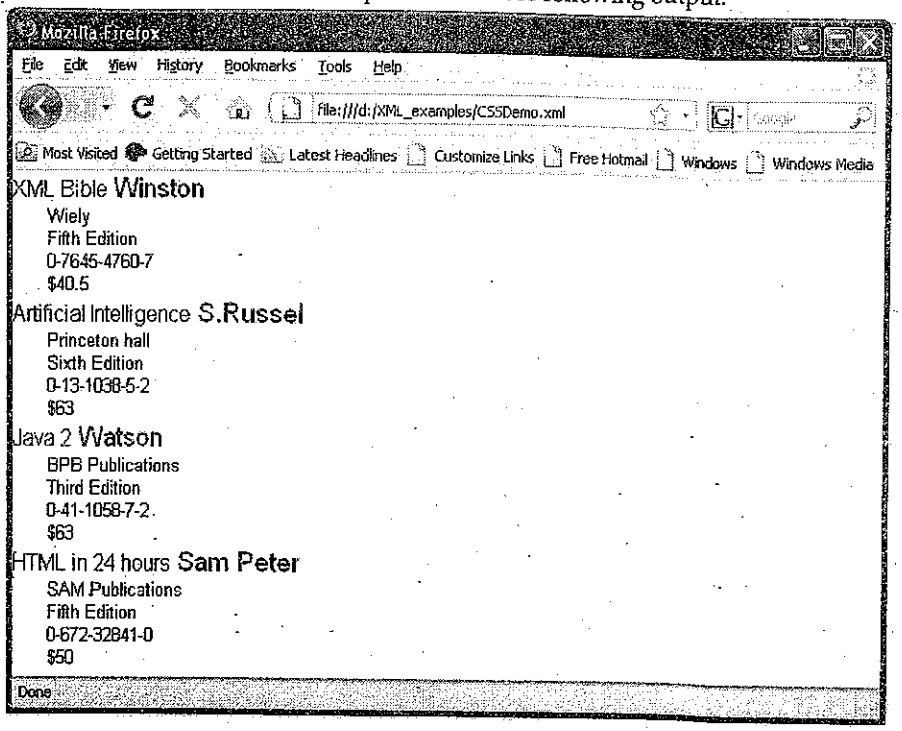
```
    <Book>
        <Title>HTML in 24 hours</Title>
        <Author>Sam Peter</Author>
        <Publication>SAM Publications</Publication>
        <Edition>Fifth Edition</Edition>
        <ISBN>0-672-32841-0</ISBN>
        <Price>$50</Price>
    </Book>
</Catalog>
```

Step 2 : Now we will write a simple CSS file using notepad or some suitable text editor. This file has the extension .css and it describes the styles of various tags.

CSS file

```
Catalog
{
font-family:arial;
color:red;
font-size:16pt
}
Book
{
display:block;
font-family:times new roman;
color:blue;
font-size:14pt
}
Title
{
font-family:arial;
color:green;
font-size:12pt
}
Author
{
font-family:arial;
color:magenta;
}
Publication,Edition,ISBN,Price
{
display:block;
font-family:arial;
color:black;
font-size:10pt;
margin-left: 20pt
}
```


Step 3 : Go to some web browser like Internet Explorer or Mozilla Firefox and type the name of XML file created in step 1. You can see following output.



Review Question

1. *Why would you use CSS style sheet for an XML document ? Illustrate with example.*

May-12, Dec.-12, Marks 10

4.9 XSLT

The XSLT stands for **XSL Transformations** and XSL stands for **eXtensible Stylesheet Language**. An XSLT is a W3C recommendation. The XSLT is used for defining the XML document transformations and presentations. Hence XSL decides how an XML document should look on the web browser. XSL consists of three parts :

1. XSLT is a language for transforming XML documents.
2. XPath is a language for navigating in XML documents.In other words we can reach to any node of XML document using XPath.
3. XSL-FO is a language for formatting XML documents for displaying it in desired manner.

4.9 Overview of XSLT

- Using XSLT we can decide the way by which we want the element to get displayed.
- XSLT uses **XPath** to find information in an XML document. XPath is used to navigate through elements and attributes in XML documents. We can sort these elements, hide or display particular element and can apply some decision making logic on these elements.
- It is a function-style programming language. The syntactic structure of XSLT is XML. Hence each statement is specified using the element. The XSLT works in following manner -
- XSLT Processors **take two input documents** one is **XML document** and another is **XSLT document.** The XSLT document is nothing but a program and XML document is nothing but the input data, thus this program works on the XML input data. Then some or whole part of the XML document is selected,modified and merged with XSLT program document in order to produce another document.
- This newly produced document is provided as input to the XSLT processor which in turn produce another document called the **XSL document.**
- The XSL document is used along with the application so that particular application can be displayed on the web browser in some desired manner.

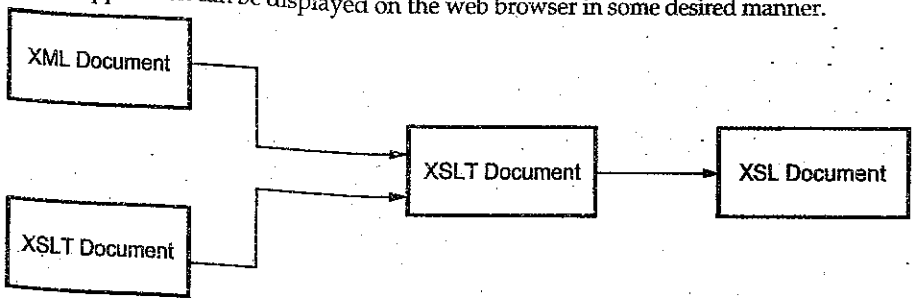


Fig. 4.9.1 Processing of XSLT document

- The XSLT document makes use of **templates** using which particular code can be described in XML document.
- XML document and then particular code in XSLT is executed when the match is found.
- XSLT processor processes the XML document sequentially by reading each line one by one.
- The XSLT model can be described as **template driven** or **data driven** model.

Let us now discuss the use of XSLT with the help of programming examples -

4.9.2 How to Create XSL Stylesheet ?

We will follow the following steps in order to create our first the XSL stylesheet.

Step 1 : Write a simple XML document. One sample example is as below -

XML Document[SimpleXml.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<Student>
 <Person-Details>
   <name>Anand</name>
   <address>Pune</address>
   <std>Second</std>
   <marks>70 percent</marks>
 </Person-Details>
 <Person-Details>
   <name>Anuradha</name>
   <address>Chennai</address>
   <std>Second</std>
   <marks>90 percent</marks>
 </Person-Details>
 <Person-Details>
   <name>Archana</name>
   <address>Mumbai</address>
   <std>Forth</std>
   <marks>80 percent</marks>
 </Person-Details>
 <Person-Details>
   <name>Monika</name>
   <address>Delhi</address>
   <std>Tenth</std>
   <marks>77 percent</marks>
 </Person-Details>
</Student>
```

Step 2 : Now create a XSL stylesheet. Note that this document **must have** file name extension as **.xsl**

XSLT Document[SimpleXml.xsl]

```
<?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>Students Database</h2>
 <table border="1">
  <tr bgcolor="gray">
   <th>Name</th>
   <th>Address</th>
   <th>Standard</th>
   <th>Marks</th>
  </tr>
  <xsl:for-each select="Student/Person-Details">
  <tr bgcolor="pink">
   <td> <xsl:value-of select="name"/> </td>
   <td> <xsl:value-of select="address"/> </td>
   <td> <xsl:value-of select="std"/> </td>
   <td> <xsl:value-of select="marks"/> </td>
  </tr>
  </xsl:for-each>
 </table>
 </body>
 </html>
</xsl:template>
</xsl:stylesheet>
```

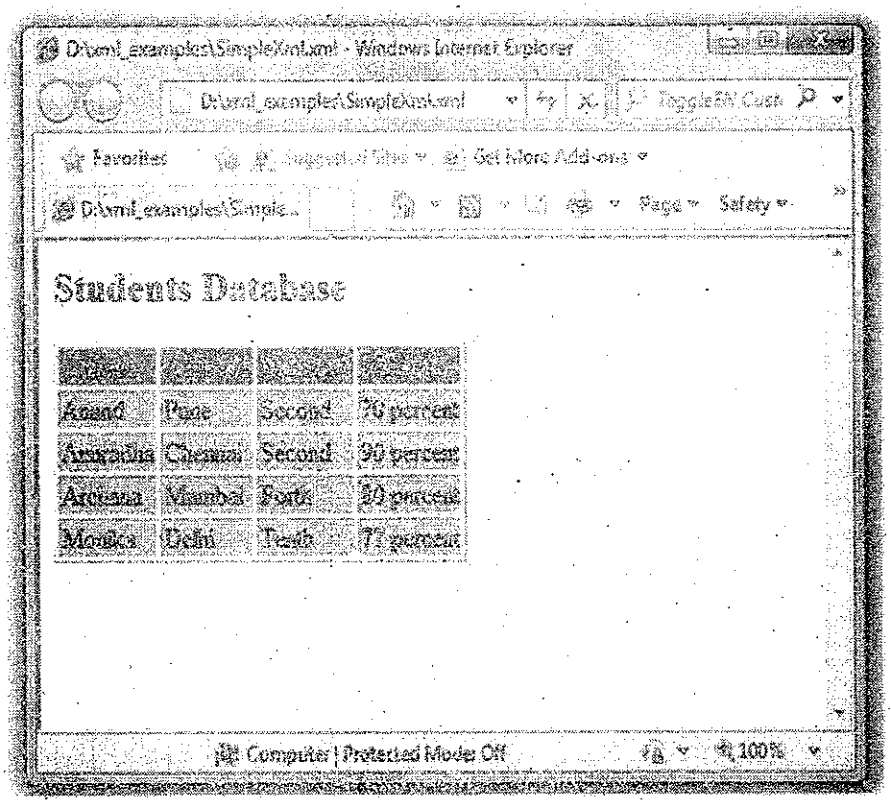
Step 3 : Now open the XML document SimpleXml.xml and add one line in it. This modification is shown by boldfaced code -

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="SimpleXml.xsl"?>
<Student>
 <Person-Details>
   <name>Anand</name>
```



```
   <address>Pune</address>
   <std>Second</std>
   <marks>70 percent</marks>
 </Person-Details>
...
```

Step 4 : For getting the result open the SimpleXml.xml file in Internet Explorer -



Script Explanation

In step 1 we have created an XML document. This document contains various elements such as Student,Person-Details,name,add,std and marks. These elements are actually accessed by an XSL file which we have written in step 2. As the XSL file is basically an XML document it must begin with the line

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

To relate an XSL file with an XML document we must write the following code in the XML document

```
<?xml-stylesheet type="text/xsl" href="SimpleXml.xsl"?>
```

Then observe the XSLT document written in step 2, it can be described as below -

1. At the top of this document we have written

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

It means that this is an XSL style sheet with version 1.0 which points to an official W3C XSLT namespace using the statement.

```
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
```

2. The XSL makes use of templates in order to match the elements of XML document. In order to match the root node of XML we use "/" .Hence the statement becomes -

```
<xsl:template match="/">
```

We can alternatively specify the root element in the match expression as follows -

```
<xsl:template match="Student">
```

Thus template is used to build the templates in XSL file.

3. Then we can make use of XHTML tags for deciding the presentation style. And then we can access the desired element of the XML document using the prefix xsl as follows -

```
<xsl:for-each select="Student/Person-Details">
```

The XSL **<xsl:for-each>** element can be used to select every XML element of a specified node-set. Hence under the root node **Student** there are the nodes named **Person-Details** which get selected each time.

4. Then using the **select** clause of **value-of** element we can extract the value of the desired element. The name of the element (written in XML document) must be assigned to the select clause.

In above application the table header has the background color grey and the each cell in the table has the background color pink.

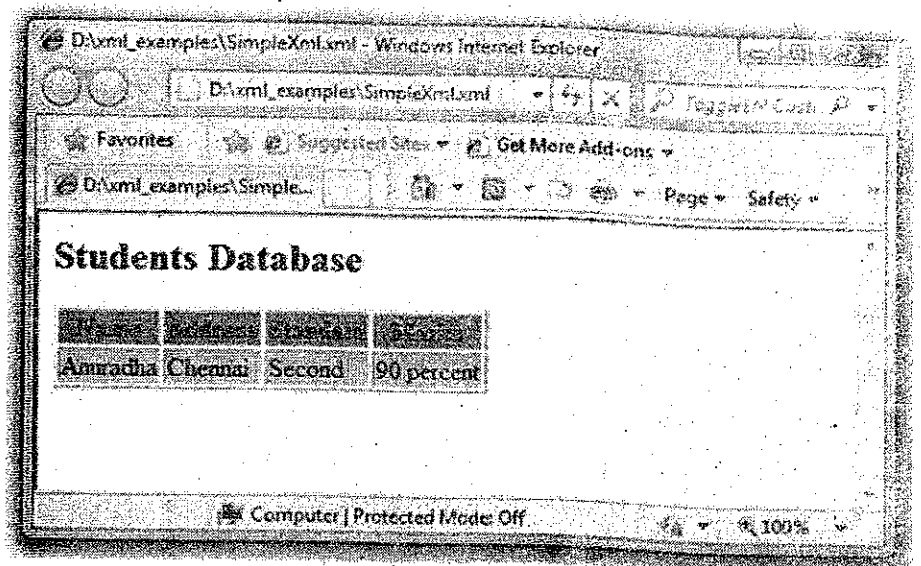
We can also filter the output from the XML file by adding a criterion to the select attribute in the <xsl:for-each> element. For example we can modify the xsl file as -

```
<?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>Students Database</h2>
 <table border="1">
  <tr bgcolor="gray">
   <th>Name</th>
   <th>Address</th>
   <th>Standard</th>
   <th>Marks</th>
  </tr>
  <xsl:for-each select="Student/Person-Details[name='Anuradha']">
  <tr bgcolor="pink">
   <td><xsl:value-of select="name"/></td>
   <td><xsl:value-of select="address"/></td>
   <td><xsl:value-of select="std"/></td>
   <td><xsl:value-of select="marks"/></td>
  </tr>
  </xsl:for-each>
 </table>
 </body>
 </html>
</xsl:template>
</xsl:stylesheet>
```

Output

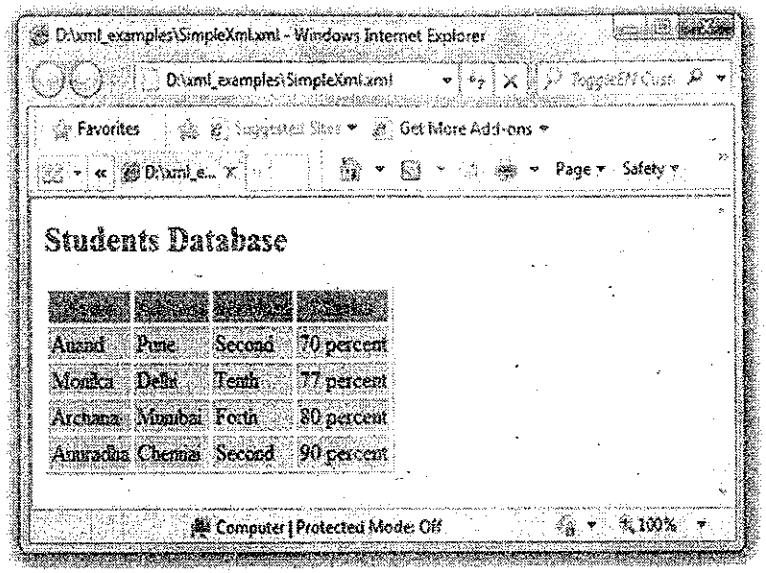


We can also display the sorted data using <xsl:sort>. Just modify your SimpleXml.xsl file as follows for displaying the sorted data. The sorting is based on the marks field -

SimpleXml.xsl

```
<?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>Students Database</h2>
 <table border="1">
  <tr bgcolor="gray">
   <th>Name</th>
   <th>Address</th>
   <th>Standard</th>
   <th>Marks</th>
  </tr>
  <xsl:for-each select="Student/Person-Details">
  <xsl:sort select="marks"/>
  <tr bgcolor="pink">
   <td><xsl:value-of select="name"/></td>
   <td><xsl:value-of select="address"/></td>
   <td><xsl:value-of select="std"/></td>
   <td><xsl:value-of select="marks"/></td>
  </tr>
  </xsl:for-each>
 </table>
 </body>
 </html>
</xsl:template>
</xsl:stylesheet>
```


The output can then be -



Notes

5 Introduction to PHP

Syllabus

Origins and Uses of PHP, Overview of PHP, General Syntactic Characteristics, Primitives, Operations and Expressions, Variables, Integer Type, Double Type, String Type, Boolean Type, Arithmetic Operations & Expressions, String Operations, Scalar Type conversions, Output, Control statements, Relational Operators, Boolean Operators, Selection Statements, Loop statements, An Example, Arrays, Array Creation, Accessing array Elements, Functions for Dealing with Arrays, Sequential Access to Array Elements, Sorting Arrays, Functions, General Characteristics of Functions, Parameters, The scope of Variables, The Lifetime of Variables, Pattern Matching, Form Handling, Files, Opening and Closing Files, Reading from a File, Writing to a File, Locking Files, Cookies, Introduction to Cookies, PHP Support for Cookies, Session Tracking.

Contents

5.1 Origins and Uses of PHP

- PHP was developed in 1994 by Apache group.
- PHP stands for PHP: Hypertext Preprocessor.
- PHP is a server-side scripting language. It is mainly used for form handling and database access.
- It is free to download and use.

5.2 Overview of PHP

- PHP is a server side scripting language embedded in XHTML. It is an alternative to CGI, ASP,ASP.NET and JSP.
- The extension to the PHP files are .php,.php3 or .phtml.
- The php processor works in **two modes**. If the PHP processor finds XHTML tags in the PHP script then the code is simply copied to the output file. But when the PHP processor finds the PHP code in the script then that code is simply interpreted and the output is copied to the output file.
- If you click for view source on the web browser you can never see the PHP script because the output of PHP script is send directly to the browser but you can see the XHTML tags.
- PHP makes use of dynamic typing that means there is no need to declare variables in PHP. The type of variable gets set only when it is assigned with some value.
- PHP has large number of library functions which makes it flexible to develop the code in PHP.

5.2.1 Installation of PHP

- There are various methods of getting PHP installed on your machine.
- The PHP requires the Apache Web server to execute its code. The Apache web server is an open source software and can be easily downloaded from the Internet. The site for getting the Apache web server is

  http://httpd.apache.org/download.cgi.
- Then PHP can be installed on you machine from the web site **http://www.php.net.**
- Another approach which is the most efficient way to install Apache,PHP,MYSQL on you machine is to use the packages like XAMPP(The X stands for any OS) or WAMPP(the W stands for Windows OS). These packages support the Apache,PHP,MUSQL,PERL.

- I have installed the XAMPP package. The root directory can be accessed using the **http://localhost/**

Review Question

1. *Write a note on PHP.* Question Bank, Marks 5

5.3 General Syntactic Characteristics

- PHP code can be embedded in the XHTML document. The code must be enclosed within **<?php** and **>**
- If the PHP script is stored in some another file and if it needs to be referred then **include** construct is used. For instance:

  Include("myfile.inc")
- The variable names in PHP begin with the $ sign.
- Following are some reserved keywords that are used in PHP.

and	default	false	if	or	this
break	do	For	include	require	true
case	else	foreach	list	return	var
class	elseif	function	new	static	virtual
continue	extends	global	not	switch	while
					xor

- The **comments** in PHP can be #,//, /* ...*/
- The PHP statements are terminated by semicolon.

How to write and execute PHP documents ?

Open some suitable text editor like Notepad and type the following code. Save your code by the extension **.php.**

It is expected that the PHP code must be stored in **htdocs** folder of Apache.

As I have installed xampp, I have got the directory c:\xampp\htdocs. I have created a folder named php-examples inside the htdocs and stored all my PHP documents in that folder.

Hence when I want to get the output of the PHP code I always give the URL

http://localhost/php-examples/programmName.php

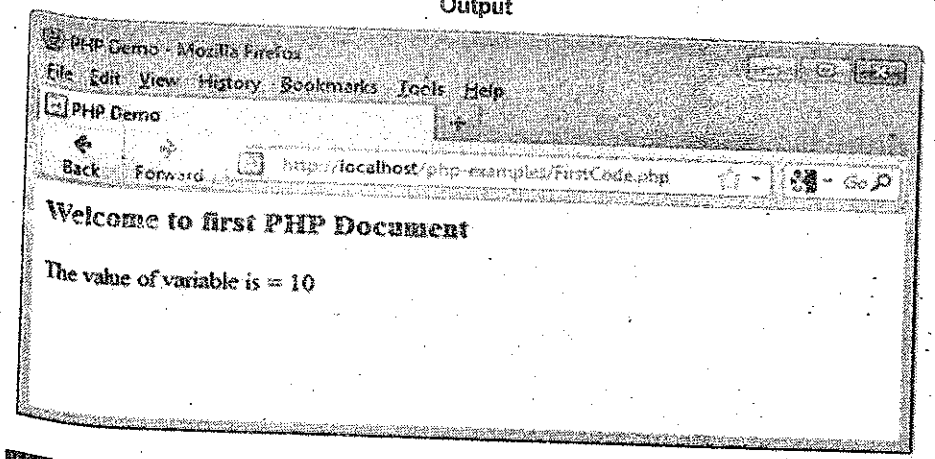
The http://localhost refers to the path **c:\xampp\htdocs**

Following is the first example of PHP script

PHP Document[FirstCode.php]

```
<html>
<head>
 <title> PHP Demo </title>
</head>
<body>
 <?php
 $i=10;
 echo"<h3>Welcome to first PHP Document</h3>";
 echo"The value of variable is = $i";
 ?>
</body>
</html>
```

Output



5.4 Primitives, Operations and Expressions

Nov.-11, Marks 3

The primitives are the basic data types used in PHP. There are four scalar types that are used in PHP and those are integer, Boolean, double and string.

5.4.1 Variables

- Variables are the entities that are used for storing the values.
- PHP is a dynamically typed language. That is PHP has no type declaration.
- The value can be assigned to the variable in following manner -

```
$variable_name=value
```


- If the value is not assigned to the variable then by default the value is NULL. The unsigned variables are called **unbound variable.**
- If the unbound variable is used in the expression then its NULL value is converted to the value 0.
- Following are some rules that must be followed while using the variables -

1. The variable must start with letter or underscore _, but it should not begin with a number.
2. It consists of alphanumeric characters or underscore.
3. There should not be space in the name of the variable.
4. While assigning the values to the variable the variable must start with the $. For example

```
$marks=100;
```

- Using the function **IsSet** the value of the variable can be tested. That means if IsSet($marks) function returns **TRUE** then that means some value is assigned to the variable **marks.**
- If the unbound variable gets referenced then the error reporting can be done with the help of a function **error_reporting(7)** .The default error reporting level is 7.

5.4.2 Integer Type

- For displaying the integer value the Integer type is used.
- It is similar to the **long** data type in C.
- The size is 32 bit.

5.4.3 Double Type

- For displaying the real values the double data type is used.
- It includes the numbers with decimal point, exponentiation or both. The exponent can be represented by E or e followed by integer literal.
- It is not compulsory to have digits before and after the decimal point. For instance .123 or 123. is allowed in PHP.

5.4.4 String Type

- There is no character data type in PHP. If the character has to be represented then it is represented using the string type itself;but in this case the string is considered to be of length 1.
- The string literals can be defined using either single or double quotes.
- In single quotes the escape sequence or the values of the literals can not be recognized by PHP but in double quotes the escape sequences can be recognized.For example

```
The total marks are= $marks
```

will be typed as it is but

```
"The total marks are= $marks"
```

will display the value of $marks variable.

5.4.5 Boolean Type

- There are only two types of values that can be defined by the Boolean type and those are TRUE and FALSE.
- If Boolean values are used in context of integer type variable then TRUE will be interpreted as 1 and FALSE will be interpreted as 0.
- If Boolean values are used in context of double type then the FALSE will be interpreted as 0.0.

5.4.6 Arithmetic Operations and Expressions

- PHP supports the collection of arithmetic operators such as +,-,/,*,%,++ and – with their usual meaning.
- While using the arithmetic operators if both the operands are integer then the result will be integer itself.
- If either of the two operands is double then the result will be double.
- PHP has large number of predefined functions. Some of these functions are enlisted in the following table -

Function	Purpose
floor	The largest integer less than or equal to the parameter is returned For example floor(4.9) will return 4.
ceil	The smallest integer less than or equal to the parameter is returned For example ceil(4.9) will return 5.
round	Nearest integer is returned.
abs	Returns the absolute value of the parameter.
min	It returns the smaller element
max	It returns the larger element

5.4.7 String Operations

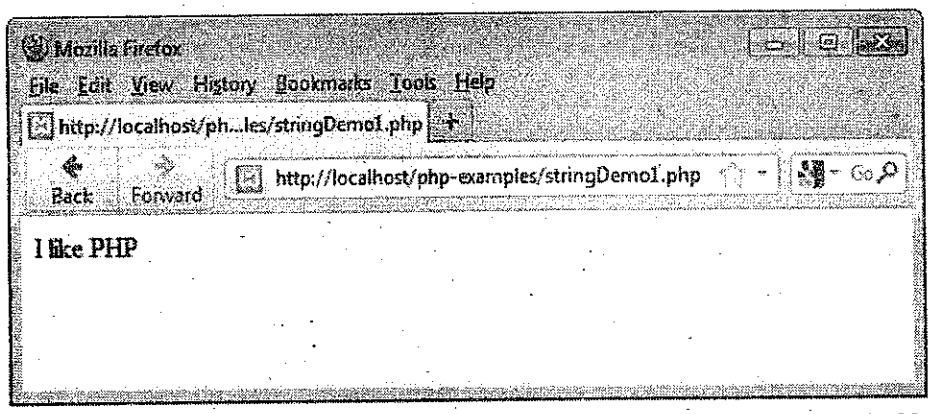
- Concatenation is the only one operator used in string.It is denoted by dot.

- Strings are treated as the array of characters. The first position of the character is indexed as 0.
- The sample PHP script that stores the string in a variable is as given below –

PHP Script[stringDemo1.php]

```
<?php
$s="I like PHP";
echo $s;
?>
```

Output



Note that the variable s is assigned with the string. The string is given in a double quotes. Then using the **echo** whatever string is stored in the variable **s** is displayed on the console.

- Various functions used for string handling are -

Function	Purpose	Sample PHP Code	Output
strlen(string1)	It finds the total number of characters in the string	<?php $s="Friend"; echo $s <?php	6
strcmp (string1,string2)	It compares the two strings. It is case sensitive. If this function returns 0 then two strings are equal. If this function returns >0 then string1 is greater than string2 If this function returns <0 then string1 is less than string2	<?php echo strcmp("PHP","PHP"); ?>	0

strtolower (string1)	This function converts the characters in string1 to lower case.	<?php echo strtolower("PHP."); ?>	php
strtoupper (string1)	This function converts the characters in string1 to upper case.	<?php echo strtolower("php"); ?>	PHP
trim(string1)	This function eliminates the white space from both the ends of the string	<?php $str = " PHP "; echo " : $str "; echo " :".trim($str); echo " "; ?>	: PHP : PHP

5.4.8 Scalar Type Conversions

- PHP supports for both implicit and explicit type conversion.
- The implicit type conversion is called **coercion**. In implicit type conversion the context of expression determines the data type that is expected or required.

Implicit conversion

- The coercion takes place between integer and double types and between Boolean and other scalar type.
- The coercion is also between the numeric and string types. That means whenever a numeric value appears in string context then it is converted to string type, similarly when the string value appears in numeric context it is automatically converted to numeric type.
- When the double value is converted to integer the fractional part is eliminated and the value is not rounded.

Explicit conversion

- There are three ways by which the explicit conversion takes place.

1. The syntax is

```
(datatype)$variable_name
```

For example -

```
(int)$marks
```

2. The syntax is -

```
Conversion_function($variable_name)
```

For example -

```
Intval($marks)
```

3. The syntax is -

```
settype($variable_name,"datatype");
```


For example -

```
settype($marks,"integer");
```

- The gettype function is useful to obtain the data type of the variable.

Review Questions

1. *Explain four scalar types of PHP.* Dec-15, Marks 5
2. *Explain built-in string manipulation functions.* Dec-16, May-17, May-18, Dec-18, Marks 5
3. *Explain Implicit and Explicit type conversions.* Dec-16, Marks 5
4. *Write a note on PHP and its scalar types.* Dec-17, Marks 10

5.5 Output

- The PHP is a server side scripting language and is used to submit the web page to the client browser. Hence it is a standard method of embedding the PHP code within the XHTML document. That means we can use the XHTML tags in PHP while displaying the output.
- The print function is used to create simple unformatted output. For example: The string can be displayed as follows

```
print "I am proud of my <b>country</b>"
```

The numeric value can also be displayed using the print. For example -

```
print(100);
```

It will display the output as 100.

- PHP also makes use of the printf function used in C. For example

```
printf("The student %d has %f marks",$roll_no,$marks);
```

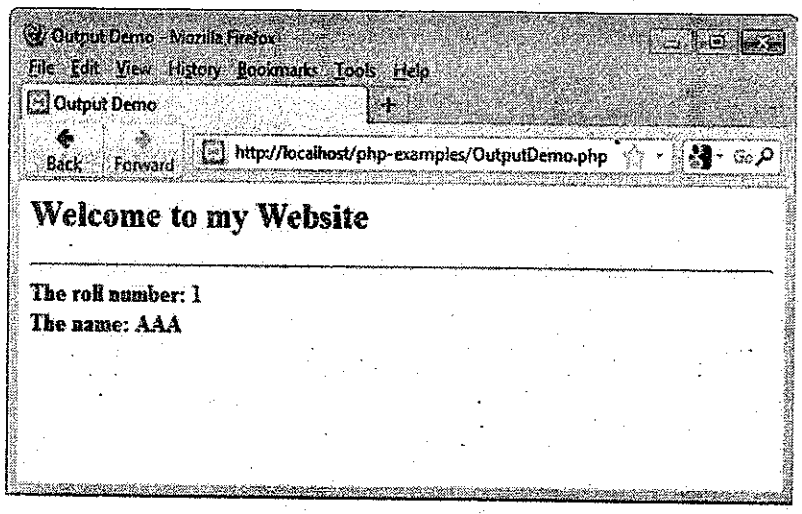
- Following is a simple PHP document which makes use of the statements for displaying the output.

PHP Document[OutputDemo.php]

```
<html>
<head>
   <title> Output Demo </title>
</head>
<body>
<?php
   print "<h2>Welcome to my Website.</h2>";
   print "<hr/>";
   $roll_no=1;
```

```
    $name="AAA";
    printf("<b>The roll number: %d</b>",$roll_no);
  print "<br/><b>";
    printf("The name: %s",$name);
    print "</b>";
?>
</body>
</html>
```

Output



5.6 Control Statements

Dec.-12, 13, May-13, Marks 5

- The control statements in PHP are similar to the control statements that are used in C.
- The control statements include if statements, while, do while and for loops.
- We will begin our discussion by understanding various operators used in PHP

5.6.1 Relational Operators

- There are eight relational operators used in PHP.
- These are <,>,<=,>=,!=,== has their usual meaning. The are six traditional operators.
- The operator === is used in PHP. It returns true if both operands that are using === have same type and have same value.
- The operator !== is opposite of ===.
- If one of the operand in the six operators is not same then the coercion will occur automatically.

5.6.2 Boolean Operators

- The Boolean operators are

Operator	Meaning
and &&	The binary AND operation is performed.
or ||	The binary OR operation is performed.
xor	The XOR operation will be performed.

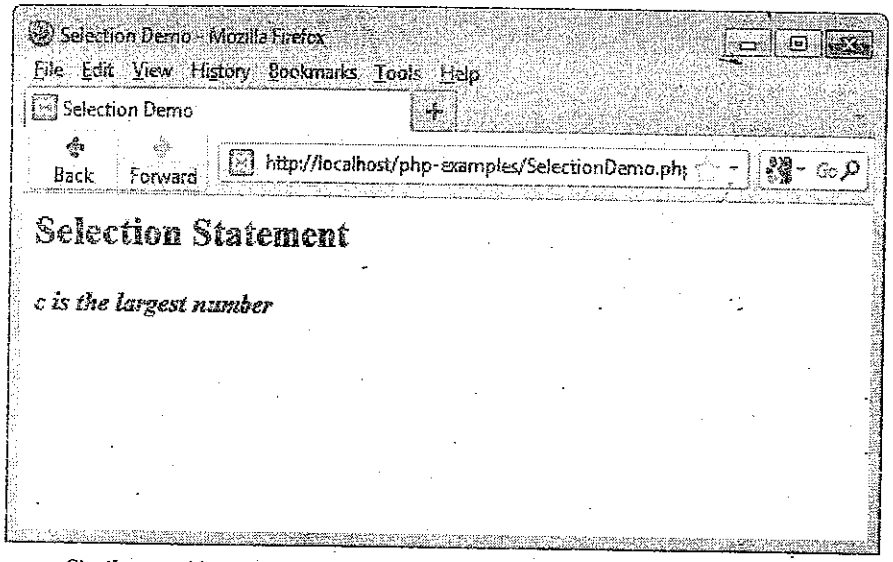
5.6.3 Selection Statements

- The **if** statement, the **if ... else** statement or **if...elseif** statements are used in selection statements. Following is an example of it

PHP Document[selection.php]

```
<html>
<head>
    <title>Selection Demo</title>
</head>
<body>
<?php
    print "<h2>Selection Statement</h2>";
    $a=10;
    $b=20;
    $c=30;
    if($a>$b)
      if($a>$c)
        print "<b> <I>a is the largest number </I></b>";
    else
        print "<b><I> c is the largest number</I> </b>";
    else
      if($b>$c)
        print "<b><I>b is the largest number</I> </b>";
    else
        print "<b><I>c is the largest number</I> </b>";
?>
</body>
</html>
```

Output

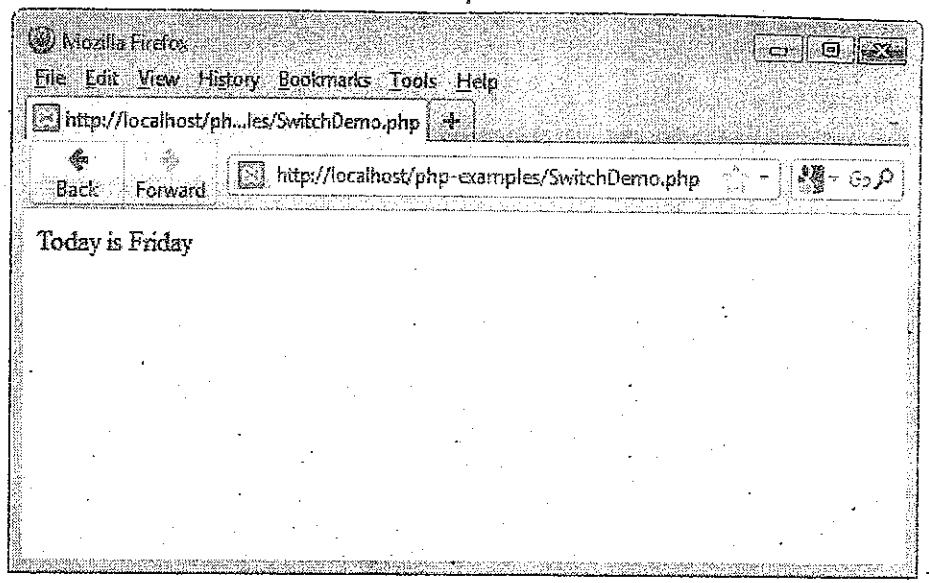


- Similar to **if** statement the **switch** statement can also be used for selection. Following is a simple PHP script for demonstrating switch statement

PHP Document[SwitchDemo.php]

```
<?php
$today = getdate();
switch($today['weekday'])
{
   case "Monday":print "Today is Monday";
                 break;
   case "Tuesday":print "Today is Tuesday";
                 break;
   case "Wednesday":print "Today is Wednesday";
                 break;
   case "Thursday":print "Today is Thursday";
                 break;
   case "Friday":print "Today is Friday";
                 break;
   case "Saturday":print "Today is Saturday";
                 break;
   case "Sunday":print "Today is Sunday";
                 break;
 default: print "Invalid input";
}
?>
```


Output



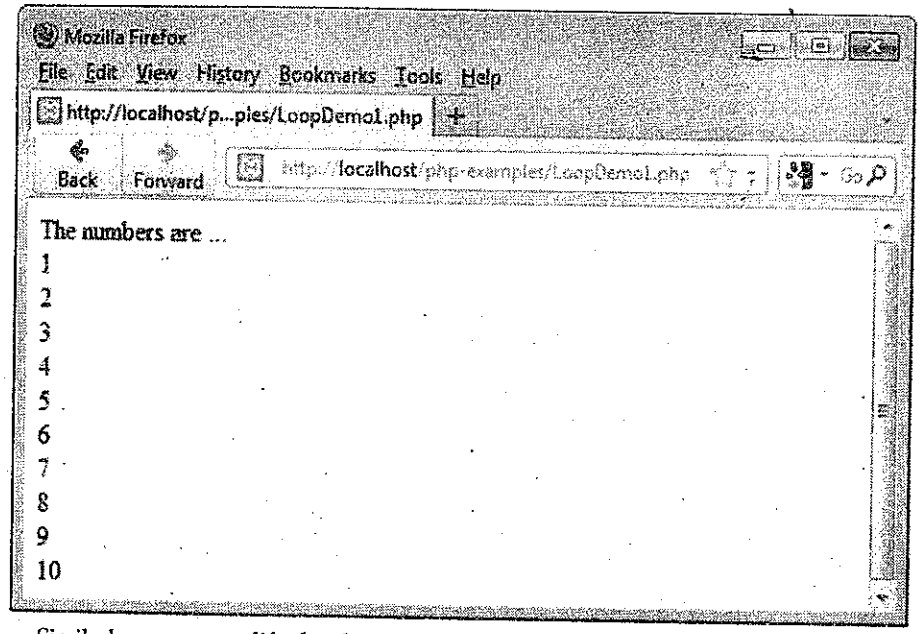
5.6.4 Loop Statements

- The **while, for** and **do-while** statements of PHP are similar to JavaScript.
- Following is a simple PHP script which displays the first 10 number

PHP Document[LoopDemo1.php]

```
<?php
$i=1;
print "The numbers are ...";
print "<br/>";
while($i<=10)
{
   print $i;
   print "<br/>";
   $i++;
}
?>
```

Output



Similarly we can modify the above script by using do-while and for loop as follows -

do ...while

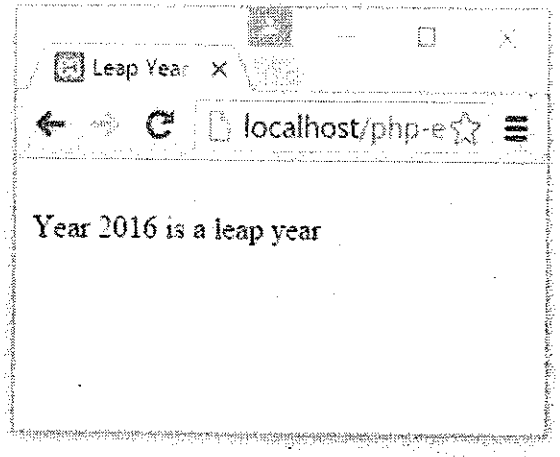
```
<?php
$i=1;
print "The numbers are ...";
print "<br/>";
do
{
    print $i;
    print "<br/>";
    $i++;
} while($i<=10);
?>
```

for

```
<?php
print "The numbers are ...";
print "<br/>";
for($i=1;$i<n)
{
    print $i;
    print "<br/>";
    $i++;
}
?>
```


5.6.5 Examples

```
<html>
<head>
<title> Sqaure and Cube Table </title>
</head>
<body>
<center>
<?php
print "<table border =1>";
print "<tr>";
print "<th>Number</th>";
print "<th>Square</th>";
print "<th>Cube</th>";
print "</tr>";
for($i=1;$i<=10;$i++)
{
print "<tr>";
print "<td>$i";
print "</td>";
print "<td>";
print $i*$i;
print "</td>";
print "<td>";
print pow($i,3);
print "</td>";
print "</tr>";
}
print "</table>";
?>
</center>
</body>
</html>
```

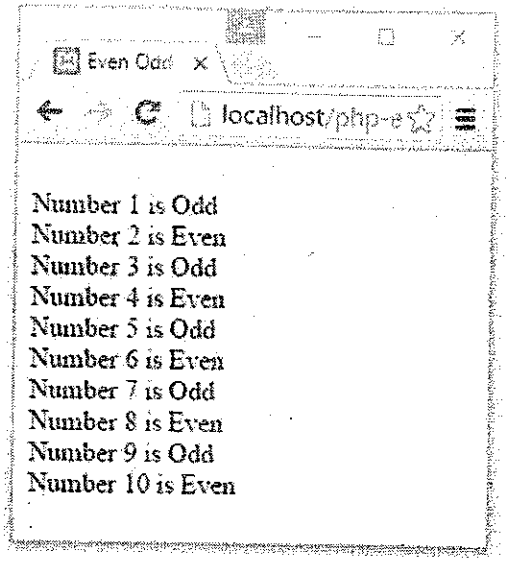


Example 5.6.2 *Write PHP programs to print whether given number is odd or even.*

Solution :

```
<html>
<head>
<title>Even Odd Demo</title>
</head>
<body>
<?php
for($i=1;$i<=10;$i++)
{
 $num=$i;
 print "<br/>";
 if($num%2==1)
  { printf("Number %d is Odd",$num); }
 else
  { printf("Number %d is Even",$num); }
}
?>
</body>
</html>
```


Output

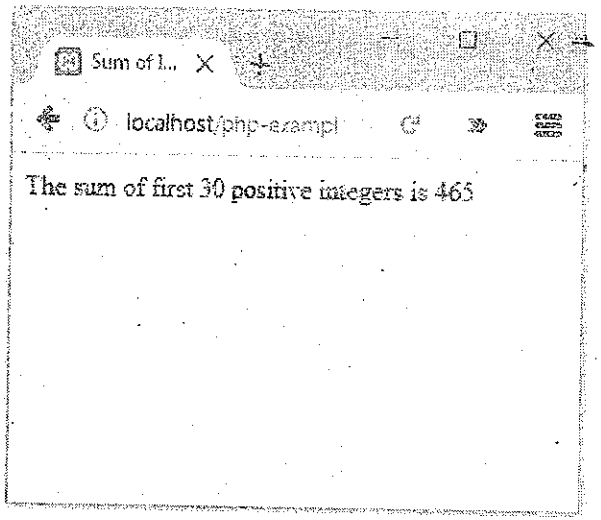


Example 5.6.3 *Write PHP script to compute the sum of positive integers upto 30 using do-while statement.* **Question Bank, Dec.-14, Marks 5**

Solution :

```
<html>
<head>
<title>Sum of Integers</title>
</head>
<body>
<?php
$sum=0;
$i=1;
do
{
 $sum=$sum+$i;
 $i++;
}while($i<=30)
 printf("The sum of first 30 positive integers is %d ",$sum);
?>
</body>
</html>
```

Output

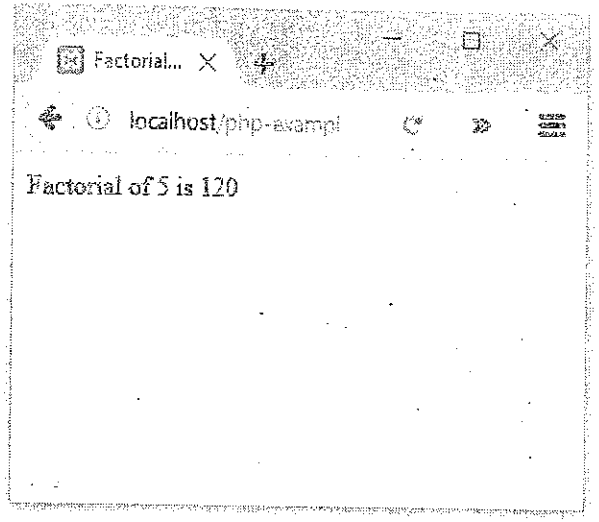


Example 5.6.4 *Write PHP script to compute factorial of 'n' using while or for loop construct* **Question Bank, Marks 5**

Solution :

```
<html>
<head>
<title>Factorial Program</title>
</head>
<body>
<?php
$n = 5;
$factorial = 1;
for ($i=$n; $i>=1; $i--)
{
 $factorial = $factorial * $i;
}
echo "Factorial of $n is $factorial";
?>
</body>
</html>
```


Output

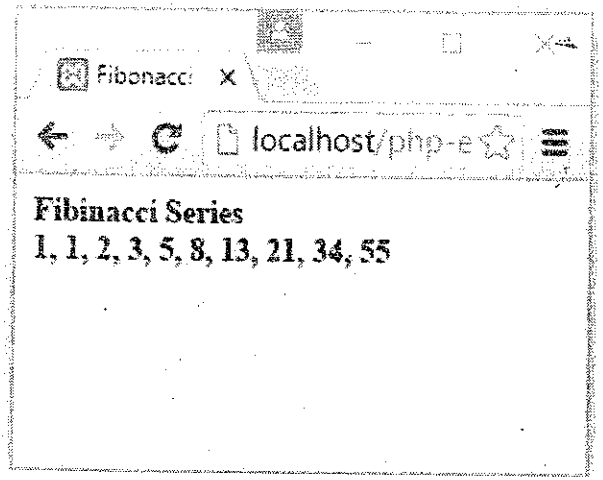


Example 5.6.5 *Write PHP script to display Fibonacci of length 10 .*

Solution :

```
<html>
<head>
<title>Fibonacci Series</title>
</head>
<body>
<?php
 $i=1;
 $j=1;
 print "<b>Fibinacci Series<br/>";
 printf("%d, %d",$i,$j);
 for($count=1;$count<9;$count++)
 {
    $k=$i+$j;
    $i=$j;
    $j=$k;
    printf(", %d",$k);
 }
?>
</body>
</html>
```

Output

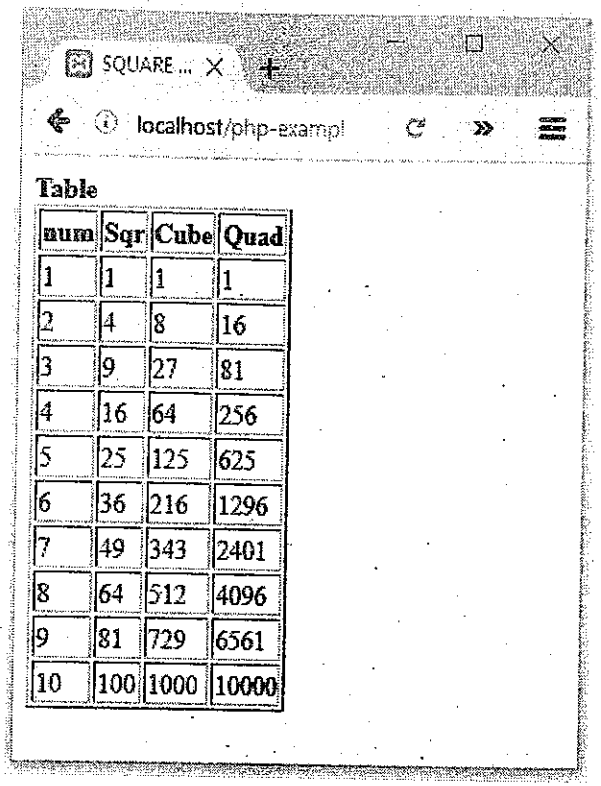


Example 5.6.6 *Construct a PHP script to compute the squareRoot, Square, Cube and Quad of 10 numbers.* **Question Bank, Marks 10**

Solution :

```
<html>
<head>
<title>SQUARE CUBE QUAD DEMO</title>
</head>
<body>
<?php

 print"<b>Table<br/>";
 print"<table border='1'>";
 print "<tr><th>num</th><th>Sqr</th><th>Cube</th><th>Quad</th></tr>";
 for($count=1;$count<=10;$count++)
 {
    $sq=$count*$count;
    $cube=$count*$count*$count;
    $quad=$count*$count*$count*$count;
    print
"<tr><td>$count</td><td>$sq</td><td>$cube</td><td>$quad</td></tr>";
 }
 print "</table>";
?>
</body>
</html>
```


Output



Example 5.6.7 *With the use of PHP, switch case and if structure perform the following and print appropriate message.*

i) Get today's date ii) If date is 3, it is dentist appointment.

iii) If date is 10, go to conference.

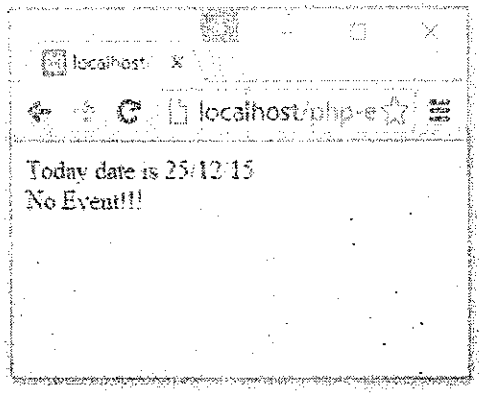
iv) If date is other than 3 and 10, no events are scheduled.

Solution :

```
<!DOCTYPE html>
<html>
<body>
<?php
    echo "Today date is " . date("d/m/y") . "<br>";
    if((date("d")<3)||(date("d")>10))
        echo "No Event!!!";
```

```
    else if((date("d")>3)&&(date("d")<10))
        echo "No Event!!!";
    else
    {
    switch(date("d"))
    {
      case 3 : echo "Dentist Appointment";
          break;
      case 10 : echo "Go to Conference";
          break;
    }
    }
?>
</body>
</html>
```

Output



Review Questions

1. *Explain the syntax of for-each statement with an example.* Question Bank, Dec. 13, Marks 5
2. *Write PHP script to illustrate loops statement* Dec. 12, Marks 5
3. *Explain the syntax of PHP switch statement with example* May 13, Marks 5

5.7 Arrays

May 13, 14, 15, 16, Nov. 11, Dec. 12, 16, Marks 10

- Arrays is a collection of similar type of elements, but in PHP you can have the elements of mixed type together in single array.
- In each PHP, each element has two parts key and value.
- The key represents the index at which the value of the element can be stored.
- The keys are positive integers that are in ascending order.

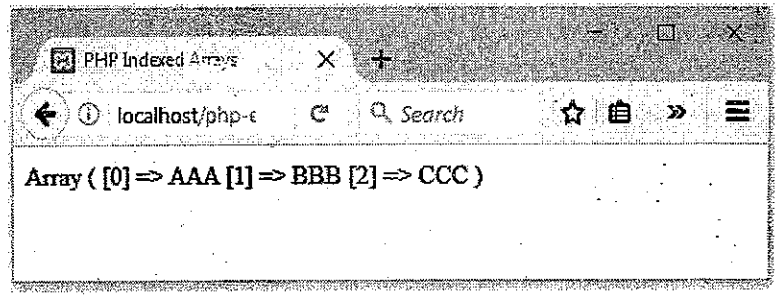
5.7.1 Array Creation

In PHP there are two types of arrays -

1. **Indexed array :** Indexed array are the arrays with numeric index. The array values can be stored from index 0. For example -

```
<html>
<head>
    <title>PHP Indexed Arrays</title>
    </head>
    <body>
<?php
$names = array("AAA", "BBB","CCC");
// Printing array structure
print_r($names);
?>
</body>
</html>
```

Output



Here values gets stored at corresponding index as follows -

```
$mylist[0]=10;
$mylist[1]=20;
$mylist[2]=30;
$mylist[3]=40;
$mylist[4]=50;
```

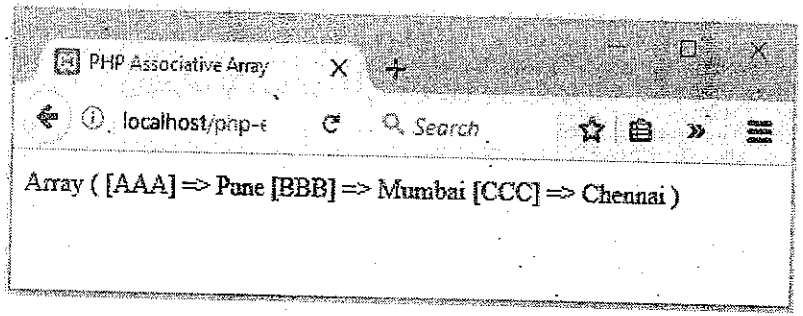
We can directly assign some value at specific index.

```
$mylist[5]=100;
```

2. **Associated array :** Associated arrays are the arrays with named keys. It is a kind of array with **name** and **value** pair.For example

```
<html>
<head>
  <title>PHP Associative Array</title>
</head>
<body>

<?php
$city["AAA"] = "Pune";
$city["BBB"] = "Mumbai";
$city["CCC"] = "Chennai";

// Printing array structure
print_r($city);
?>
</body>
</html>
```

Output



5.7.2 Accessing Array Elements

- Using an array subscript we can access the array element. The value of subscript is enclosed within the square brackets. For example -

```
$Citycode['Pune']=005;
$Name[0]="Chitra";
```

- Multiple values can be set to a single scalar variable using array. For example -

```
$people=array("Meena","Teena","Heena");
list($operator,$accountant,$manager)=$people;
```

- By this assignment Meena becomes operator, Teena becomes accountant and Heena becomes the manager.

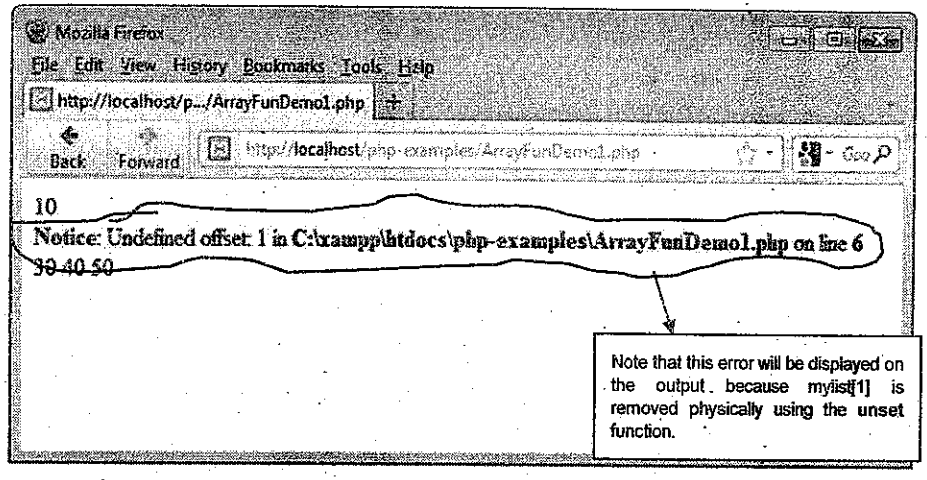
5.7.3 Functions for Dealing with Arrays

- The **unset** function is used to remove particular element from the array. For example consider following PHP document

PHP Document[ArrayFunDemo1.php]

```
<?php
$mylist=array(10,20,30,40,50);
unset($mylist[1]);
for($i=0;$i<=4;$i++)
{
   print $mylist[$i];
   print " ";
}
?>
```

Output

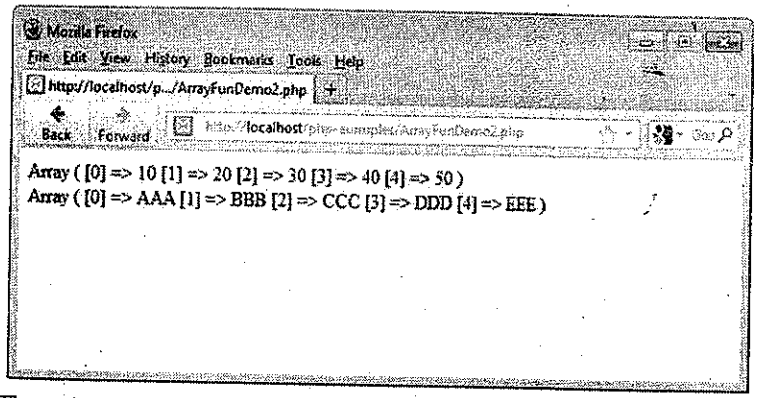


- The functions **array_keys** and **array_values** are used to return the array keys and the values at corresponding key. For example consider the following PHP document -

PHP Document[ArrayFunDemo2.php]

```
<?php
$mylist=array(10=>"AAA",20=>"BBB",30=>"CCC",40=>"DDD",50=>"EEE");
$Roll=array_keys($mylist);
$Name=array_values($mylist);
print_r($Roll);
print "<br/>";
print_r($Name);
?>
```

Output

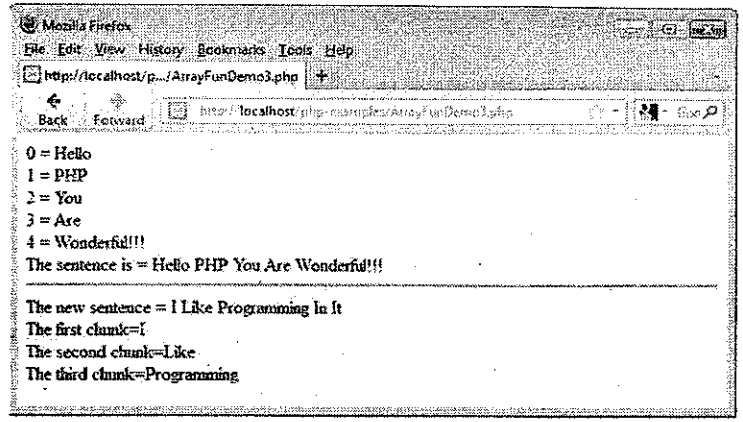


- The existence of particular key can be checked by using the **array_key_exists** function. This function returns the Boolean value.
- The **is_array** returns the Boolean value. This function takes a variable as a parameter. If the function returns TRUE then that means the parameter passed to this function is of array type.
- The **implode** and **explode functions** are used to break the word into strings or vice versa. For example consider the following PHP code -

PHP Document[ArrayFunDemo3.php]

```
<?php
$mylist = array("Hello", "PHP", "You", "Are", "Wonderful!!!");

$mySentence = implode(" ", $mylist);
for($i = 0; $i < count($mylist); $i++)
{
    echo "$i = $mylist[$i] <br />";
}
echo "The sentence is  = $mySentence <br />";
print "<br/>";
$newSentence="I Like Programming In It";
$chunks=explode(" ",$newSentence);
echo "The new sentence = $newSentence";
print "<br/>";
echo "The first chunk=$chunks[0]";
print "<br/>";
echo "The second chunk=$chunks[1]";
print "<br/>";
echo "The third chunk=$chunks[2]";
print "<br/>";
?>
```


Output



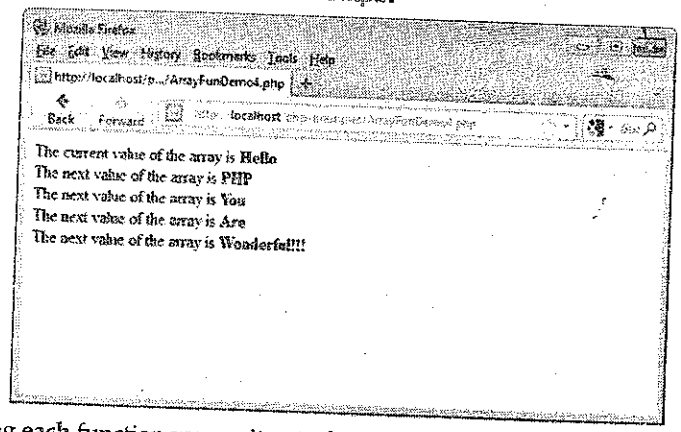
5.7.4 Sequential Access to Array Elements

- The array element reference start at the first element and every array maintains an internal pointer using which the next element can be easily accessible.
- This helps to access the array elements in sequential manner.
- The pointer **current** is used to point to the current element in the array. Using the **next** function the next subsequent element can be accessed. Following PHP code illustrates this idea -

PHP Document[ArrayFunDemo4.php]

```
<?php
$mylist = array("Hello", "PHP", "You", "Are", "Wonderful!!!");
$myval=current($mylist);
print("The current value of the array is <b>$myval</b>");
print "<br/>";
$myval=next($mylist);
print("The next value of the array is <b>$myval</b>");
print "<br/>";
$myval=next($mylist);
print("The next value of the array is <b>$myval</b>");
print "<br/>";
$myval=next($mylist);
print("The next value of the array is <b>$myval</b>");
?>
```

Output

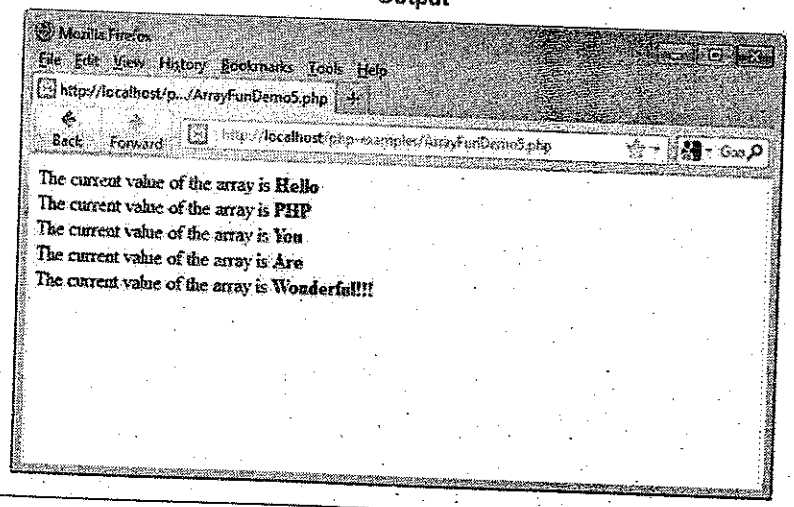


- Using each function we can iterate through the array elements.

PHP Document[ArrayFunDemo5.php]

```
<?php
$mylist = array("Hello", "PHP","You","Are","Wonderful!!!");

while($myval=each($mylist))
{
    $val=$myval["value"];
    print("The current value of the array is <b>$val</b>");
    print "<br/>";
}
?>
```

Output



- The **foreach** function is used to iterate through all the elements of the loop. The syntax of foreach statement is as follows -

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

- The above code can be modified and written as follows -

PHP Document[ArrayFunDemo6.php]

```
<?php
$mylist = array("Hello","PHP","You","Are","Wonderful!!!");

foreach($mylist as $value)
{
    print("The current value of the array is <b>$value</b>");
    print "<br/>";
}
?>
```

The output will be the same as above.

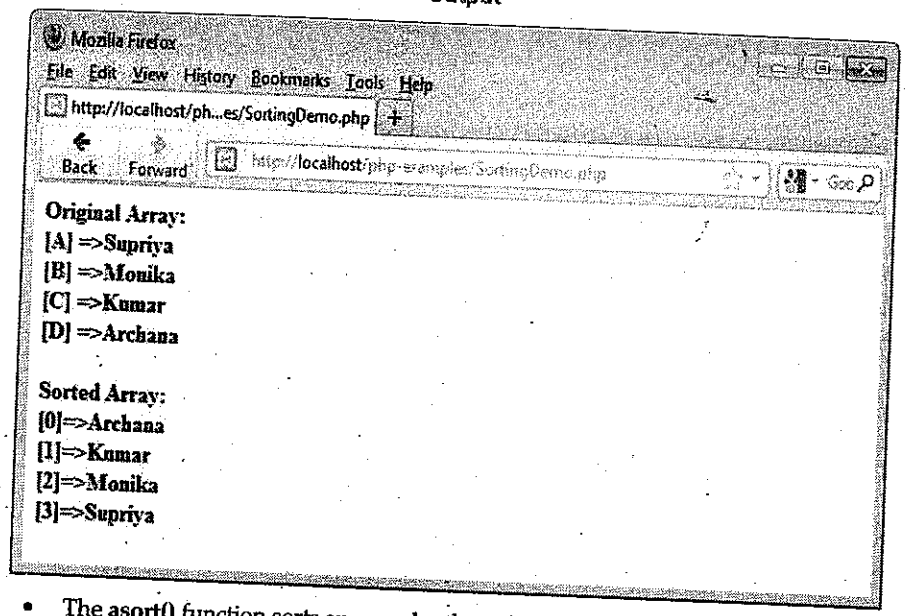
5.7.5 Sorting Arrays

- Sorting is a process in which the elements of arrays in some specific order. There are two types ordering which are followed in sorting - ascending order and descending order.
- Basically PHP uses **sort** function for sorting the array elements. There are some other functions that are also available for sorting the arrays in desired manner.
- The **sort** function sorts the array based on the values. After applying the **sort** function this function assigns new keys to the values of the array.
- Following PHP document illustrates these functions -

PHP Document[SortingDemo.php]

```
<?php
$A = array("A" => "Supriya","B" => "Monika", "C" => "Kumar","D"=>"Archana");
print "<b>Original Array:<b><br/>";
foreach($A as $key => $value )
 print(" [$key] =>$value <br/>");
sort($A);
print "<br/>";
print "<b>Sorted Array:<b><br/>";
foreach($A as $key=>$value)
 print(" [$key]=>$value <br/>");
?>
```

Output



- The **asort()** function sorts an array by the values. But the original keys are kept.
- The **ksort** function sorts the array by keys but each value's original key is retained.
- Following PHP illustrates both of these functions –

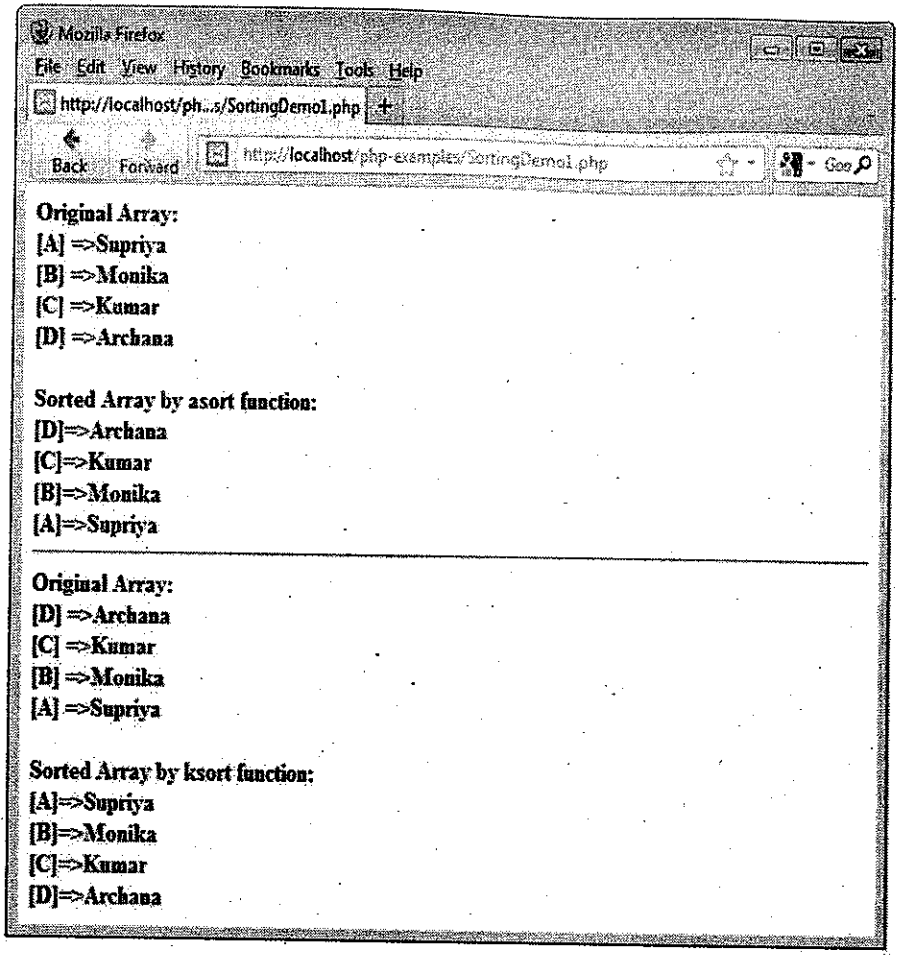
PHP Document[SortingDemo1.php]

```
<?php
$A = array("A" => "Supriya", "B" => "Monika", "C" => "Kumar", "D" => "Archana");
print "<b>Original Array: <b> <br/>";
foreach($A as $key => $value )
 print(" [$key] =>$value <br />");
asort($A);
print "<br/>";
print "<b>Sorted Array by asort function: <b> <br/>";
foreach($A as $key=>$value)
 print("[$key]=>$value <br/>");
print "<hr/>";
$B = array(30=> "CCC", 20=> "BBB", 40=> "DDD", 10=>"AAA");
print "<b>Original Array: <b> <br/>";
foreach($A as $key => $value )
 print(" [$key] =>$value <br />");
ksort($A);
print "<br/>";
print "<b>Sorted Array by ksort function: <b> <br/>";
```



```
foreach($A as $key=>$value)
 print("[$key]=>$value <br/>");
?>
```

Output



Review Questions

1. *Write a PHP script to sort an array of elements.* **Question Bank, May-13, May-16, Marks 5**
2. *Explain how to create indexed and associated array with an example.*
   Question Bank, Model Question Paper, Marks 10
3. *Write a PHP script to illustrate sort, assort and ksort functions.*
   Question Bank, Model Question Paper, May-15, Marks 10

4. *Explain types of arrays with an example in PHP.*

Question Bank, Marks 10, Nov.-11, Marks 5

5. *What is an array? In what ways can arrays in PHP is created ?* Dec.-12, Marks 5

6. *Explain functions used for sequential access to array elements with an example.* May-14, Marks 5

7. *Explain the actions of the following functions:*

i) array_keys()

ii) array_values()

iii) implode()

iv) explode()

v) array_key_exists() Dec.-16, Marks 10

5.8 Functions

May-12, 13, Marks 5

The functions in PHP are very much similar to the functions in C. Let us discuss it in detail -

5.8.1 General Characteristics of Functions

- The syntax of the function definition is as follows -

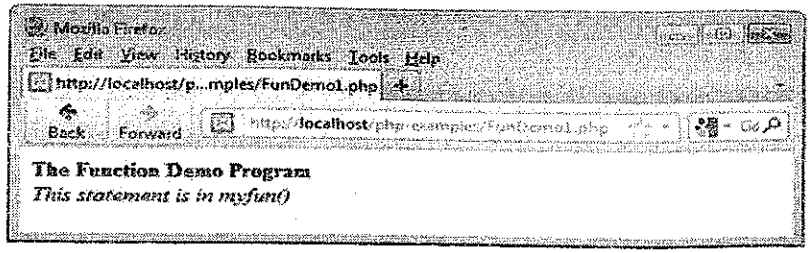
```
function name_of_function(parameter list)
{
 statements to be executed in function
...
...
...}
```

- The function gets executed only after the call to that function. The call to the function can be from anywhere in the PHP code.For example -

PHP Document[FunDemo1.php]

```
<?php
function myfun()
{
    print "<i>This statement is in myfun()</i>";
}
print "<b>The Function Demo Program</b>";
print "<br/>";
myfun();
?>
```


Output

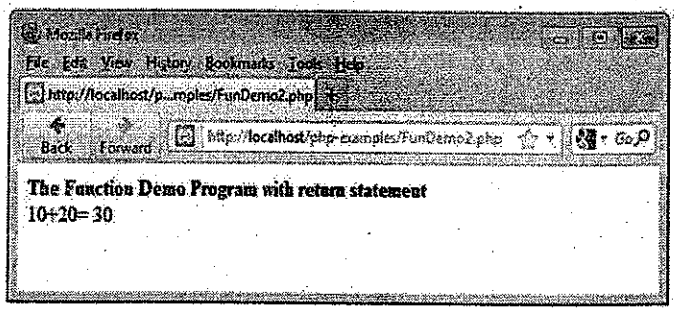


- The **return** statement is used for returning some value from the function body. Following PHP scripts shows this idea.

PHP Document[FunDemo2.php]

```
<?php
function Addition()
{
    $a=10;
    $b=20;
    $c=$a+$b;
    return $c;
}
print "<b>The Function Demo Program with return statement</b>";
print "<br/>";
print "10+20=".Addition();
?>
```

Output



5.8.2 Parameters

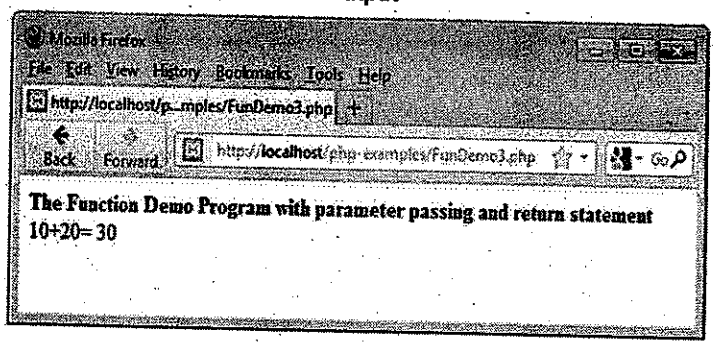
- The parameters that we pass to the function during the call is called the **actual parameter.** These parameters are generally the expressions.
- The parameters that we pass to the function while defining it is called the **formal parameters.** These are generally the variables. It is not necessary that the number of actual parameters should match with the number of formal parameters.

- If there are few actual parameter and more formal parameters then the value of formal parameter is will be some unbounded one.
- If there are many actual parameters and few formal parameters then the excess of actual parameters will be ignored.
- The default parameter passing technique in PHP is **pass by value**. The parameter passing by value means the values of actual parameters will be copied in the formal parameters. But the values of formal parameters will not be copied to the actual parameters.
- Following PHP script illustrates the functions with parameters

PHP Document[FunDemo3.php]

```
<?php
function Addition($a,$b)
{
    $c=$a+$b;
    return $c;
}
print "<b>The Function Demo Program with parameter passing and return statement</b>";
print"<br/>";
$x=10;
$y=20;
print "10+20= ".Addition($x,$y);
?>
```

Output



There are two ways to pass parameters by reference.

1. **Add & at the beginning of the name of the formal parameter.** For example -

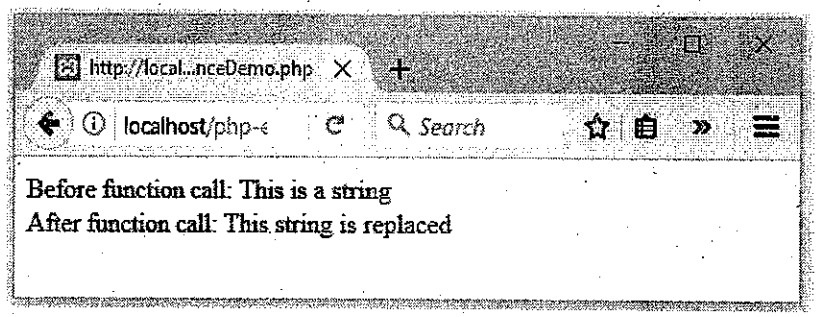
```
<?php
function add_some_extra($string)
{
  $string= 'This string is replaced';
}

$str = 'This is a string';
$str1=&$str;  //adding & at the beginning of the name of formal parameter.
print "Before function call: $str<br/>";
add_some_extra($str1);
print "After function call: $str<br/>";
?>
```


2. **Add & to actual parameter in the function call.** For example –

```
<?php
function add_some_extra(&$string)  //adding & to actual parameter in function call
{
  $string= 'This string is replaced';
}

$str = 'This is a string';
print "Before function call: $str<br/>";
add_some_extra($str);
print "After function call: $str<br/>";
?>
```

The output of the above code is one and the same it will be as follows -

Output



5.8.3 The Scope of Variables

- In PHP the scope of the variable is **local**. That means we can define a variable within a function and by the same name another variable can be defined outside the function. These two variables are considered to be different entities.
- Example :

```
<?php
function myfun()
{
    $a=10;
```

```
    print "The value of a=$a";
}
myfun();
print "<br/>";
$a=20;
print "The value of a=$a";
?>
```

Output

```
The value of a=10
The value of a=20
```

- We can define the **global variable** also. Following is the PHP script -

PHP Document

```
<?php
$a=10;
function myfun()
{
    global $a;
    $a+=10;
    print "The value of a=$a";
}
myfun();
print "<br/>";
$a=$a-5;
print "The value of a=$a";
?>
```

Output

```
The value of a=20
The value of a=15
```

5.3.4 The Life Time of Variables

- The lifetime of a variable cab be defined as the time from which it is used first to the end of function execution.
- In PHP the **static variable** is used to remember the previous values.
- The lifetime of static variable is the time when the variable if first used and it ends when the script terminates its execution.
- For the declaration of the static variable the keyword **static** is used. For example -

```
function myfun()
{
 static $count=0;
 $count++; print "The number of times you have visited this page is $count<br/>";
}
```


Review Questions

1. *What is function ? What are two ways you can specify that parameter is to be paved by reference ?* May 12, Marks 5
2. *Explain how to define user defined function in PHP with an example* May 13, Marks 5

5.9 Pattern Matching

Dec. 13, 14, May 14, 16, Marks 5

1. **preg_match function:** The preg_match() function searches string for pattern, returning true if pattern exists, and false otherwise.

Syntax

```
int preg_match ( string $pattern , string $subject [, array &$matches [, int $flags = 0 [,
int $offset = 0 ]]] )
```

where

pattern denotes the pattern to be searched for

subject denotes the input string

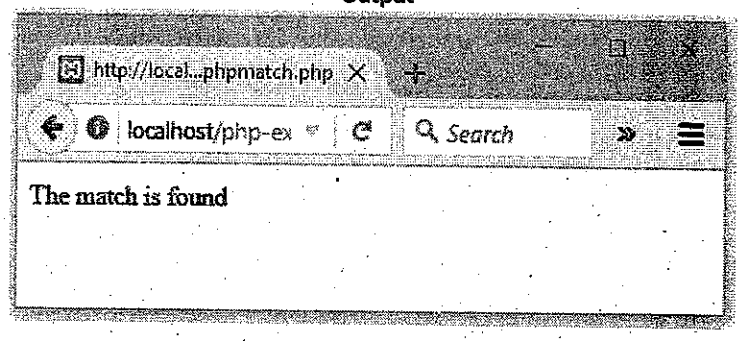
$matches[0] will contain the text that matched the full pattern

The optional parameter offset can be used to specify the alternate place from which to start the search.

Example

```
<?php
if(preg_match("/gram/","I like programming in PHP"))
    print "The match is found";
else
    print "The match is not found";
?>
```

Output



2. **preg_split function:** This function splits the string into chunks

Syntax

```
array preg_split (string $pattern , string $subject [, int $limit = -1 [, int $flags = 0 ]] )
```

where

The **pattern** is to search for, as a string.

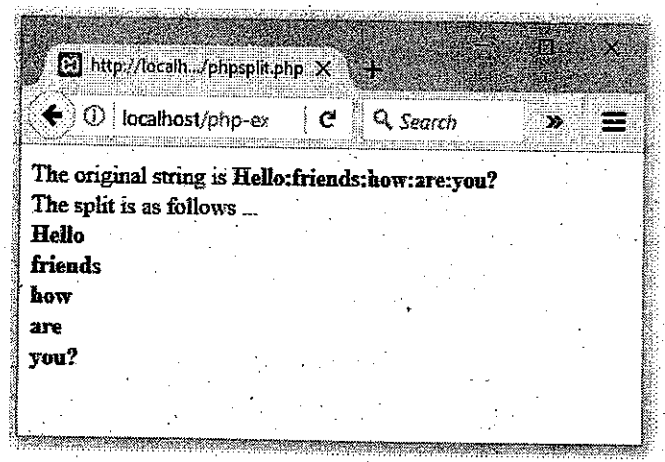
Subject: is the input string.

limit : If specified, then only substrings up to limit are returned with the rest of the string being placed in the last substring. A limit of -1, 0 or NULL means "no limit"

Example

```
<?php
$text = "Hello:friends:how:are:you?";
print "The original string is <b> $text </b>";
 print "<br/>";
 print "The split is as follows ...<br/>";
$chunks=preg_split("/:/",$text);
while($myval=each($chunks))
{
    $val=$myval["value"];
    print("<b>$val</b>");
    print "<br/>";
}

?>
```

Output



3. preg_replace function : This function replaces the given pattern from the text by the replacement string.

Syntax

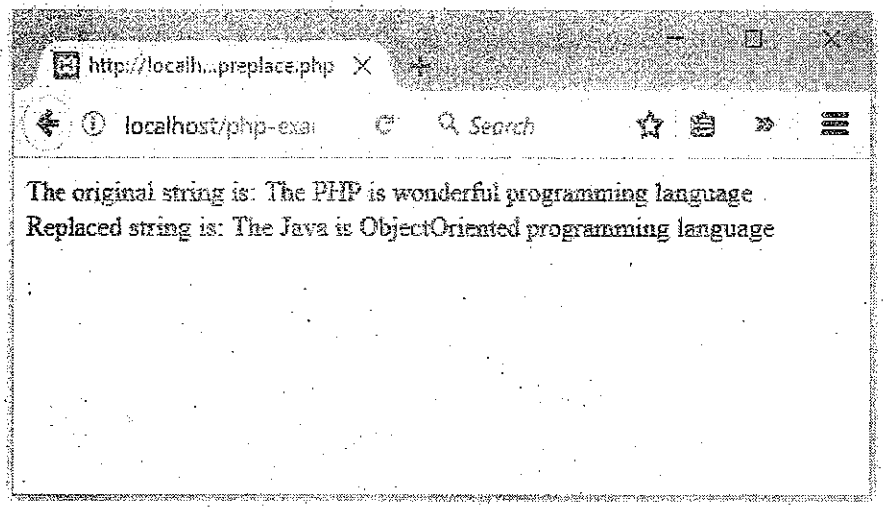
```
preg_replace ( $pattern , $replacement , $subject)
```

Where this function Searches subject for matches to pattern and replaces them with *replacement*.

Example

```
<?php
$string = 'The PHP is wonderful programming language';

$patterns = array();
$patterns[0] = '/PHP/';
$patterns[1] = '/wonderful/';

$replacements = array();
$replacements[0] = 'Java';
$replacements[1] = 'ObjectOriented';
print "The original string is: $string";
print"<br/> Repleced string is: ";
echo preg_replace($patterns, $replacements, $string);
?>
```

Output



4. **ereg function :** The ereg() function searches a string specified by string for a string specified by pattern, returning true if the pattern is found, and false otherwise.

Syntax

int ereg(string pattern, string originaltext);

Example

```
<?php
 $text = "My website is www.havefun.com";
 $retval = ereg("(\.)(com$)", $text);

 if( $retval == true )
 {
  echo "Pattern in present<br>";
 }
 else
 {
  echo "Pattern in not present<br>";
 }
```

5. **ereplace function :** The ereg_replace() function searches for string specified by pattern and replaces pattern with replacement if found.

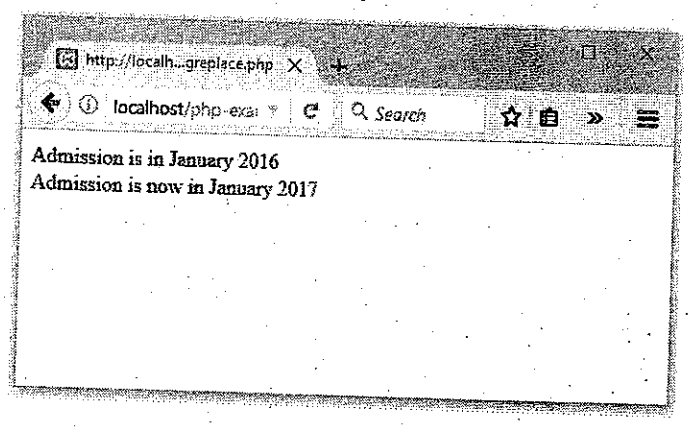
Syntax

string ereg_replace (string pattern, string replacement, string originaltext);

Example

```
<?php
 $admission_yr = "January 2016";
 print "Admission is in $admission_yr <br/>";
 $admission_yr = ereg_replace("([0-9]+)", "2017", $admission_yr);
 print "Admission is now in $admission_yr";
?>
```

Output



Review Questions

1. *Explain preg_match() and preg_split() functions with example.* Question Bank, Dec-14, Marks 6
2. *Explain preg_match() and preg_replace() function with example.* Dec-15, May-16, Marks 6
3. *Explain ereg(), ereg_replace() functions with syntax and respective example.* May-13, Marks 6

5.10 Form Handling

Dec-12, 13, May-12, 14, 16, Marks 6

- PHP is used for form handling. For that purpose the simple form can be designed in XHTML and the values of the fields defined on the form can be transmitted to the PHP script using GET and POST methods.
- For forms that are submitted via "GET" method, we can obtain the form via the $_GET array variable.
- For forms that are submitted via "POST" method, we can obtain the form via the $_POST array variable.

Example 5.10.1 *Create a form containing information sl.no, title of the book, publishers, quantity,price read the data from the form and display it using PHP script.*

Solution :

Step 1 : Create an HTML page for inputting the data. Following is the code for HTML script .

HTML Document[input.html]

```
<html>
<head>
 <title>Book Information</title>
</head>
 <body>
 <?php
$BName=$_POST['BName'];
$PUBName=$_POST['PUBName'];
$Price=$_POST['Price'];
$Qty=$_POST['Qty'];
?>
 <center>
 <h3> Book Data </h3>
 <table border=1>
 <tr>
  <th>Book name</th>
  <th>Publisher</th>
  <th>Price</th>
  <th>Quantity</th>
```

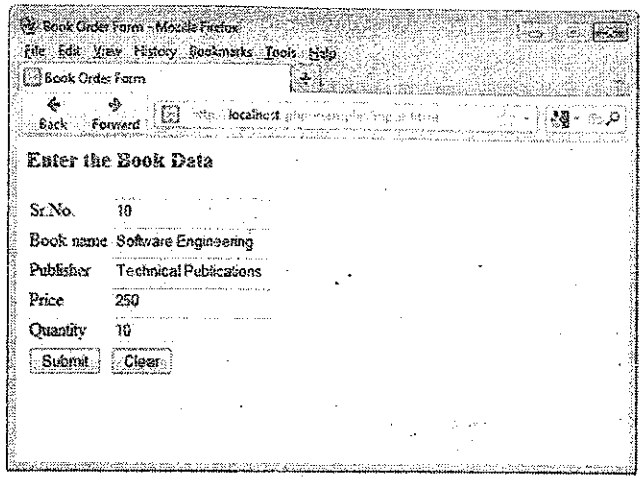
```
</tr>
<tr>
<td><?php print ("$BName"); ?></td>
<td><?php print ("$PUBName"); ?></td>
<td><?php printf("%3.2f",$Price); ?></td>
<td><?php printf("%d",$Qty); ?></td>
</tr>
</table>
</center>
</body>
</html>
```

Step 2 : Create a PHP script which will read out the data entered by the user using HTML form. The code is as follows -

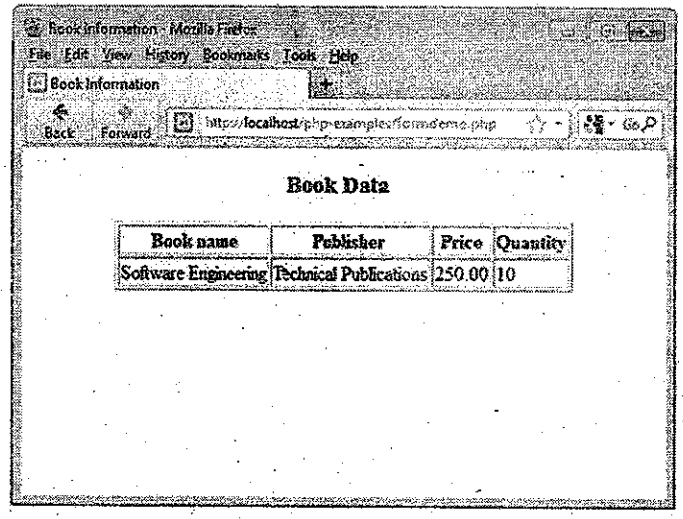
PHP Document[formdemo.php]

```
<html>
<head>
<title>Book Information</title>
</head>
<body>
<?php
$BName=$_POST['BName'];
$PUBName=$_POST['PUBName'];
$Price=$_POST['Price'];
$Qty=$_POST['Qty'];
?>
<center>
<h3> Book Data </h3>
<table border=1>
<tr>
<th>Book name</th>
<th>Publisher</th>
<th>Price</th>
<th>Quantity</th>
</tr>
<tr>
<td><?php print ("$BName"); ?></td>
<td><?php print ("$PUBName"); ?></td>
<td><?php printf("%3.2f",$Price); ?></td>
<td><?php printf("%d",$Qty); ?></td>
</tr>
</table>
</center>
</body>
</html>
```


Step 3 : Open some suitable web browser and enter the address for the HTML file which you have created in step 1.



Now click on the Submit button and the PHP file will be invoked. The output will then be as follows -



Review Question

1. *Explain post method of submitting form data with example* **Dec-13, May-16, Marks 5**

5.11 Files

PHP is known as a server-side scripting language. Hence file handling functions such as create, read, write , append are some file related operations that are supported by PHP.

5.11.1 Opening and Closing Files

- The first step in file handling is opening of the file.
- It takes two parameters - The first parameter of this function contains the name of the file to be opened and the second parameter specifies in which mode the file should be opened.

Modes	Description
r	Read only. Starts reading from the beginning of the file.
r+	Read/Write. Starts reading from the beginning of the file.
w	Write only. Opens and clears the contents of file; or creates a new file if it is not created.
w+	Read/Write. Opens and clears the contents of file; or creates a new file if it is not created.
a	Append. Opens and writes to the end of the file or creates a new file if it is not created.
a+	Read/Append. Preserves file content by writing to the end of the file.

For example :

```
$my_file = 'file.txt';
$file_handle = fopen($my_file, 'a') or die('Cannot open file: '.$my_file);
```

The **fopen** function returns TRUE if the required file is opened.

5.11.2 Reading from File

- The **fread** is the function which is used to read the file.
- It takes two parameters. The first parameter is the handle to the file and the second parameter is the number of bytes to be read. The **filesize** is the function which takes the filename as the parameter. For example -

```
$mystring = fread($file_handle, filesize("file.txt"));
```

- There is another function named **file_get_contents** using which the contents of the file can be obtained.
- The **fgets()** function is used to read a single line from the file. For example, following code displays the contents of the file line by line. –

```
while(!feof($file_handle))
{
    echo fgets($file_handle). "<br />";
}
```


Example 5.11.1 *Write PHP code to read the "Myfile.txt" line by line.*

Nov. 11, Marks 5

Solution :

Step 1 : Create a input file Myfile.txt as follows

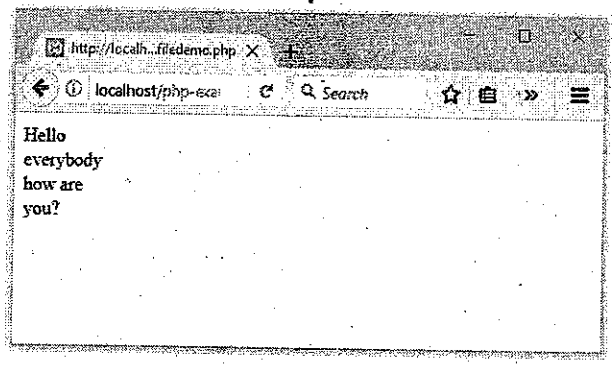
```
Hello
everybody
how are
you ?
```

Step 2 : Create a PHP script for reading the input file line by line as follows –

Filedemo.php

```
<?php
$file = fopen("Myfile.txt","r");
while(! feof($file))
{
 echo fgets($file). "<br />";
}
fclose($file);
?>
```

Output



5.11.3 Writing to a File

- The **fwrite** is the function which is used to write the contents to the file.
- It takes two parameters - the first parameter is the handle to the file and the second parameter is the number of bytes to be written. For example -

```
$Written_string = fread($file_handle, $my_data);
```

5.11.4 Locking Files

Multiple PHP scripts can access the same file at a time. But this causes a conflict problems. That means - there can be the situation in which one script is reading the file and at the same time other script is writing to text file. Sometimes there can be a situation in which two scripts are trying to write different data to the same file. These are totally undesirable in file handling technique.

The solution to this problem is to lock the file when one script is accessing it. Due to locking of the file simultaneous access to the same file by two different scripts can be avoided.

The php uses flock function for locking the files

Syntax of flock is

flock(file,lock,block)

where

file is the name of the file which need to be accessed.

lock is a kind of lock being used. Possible values are

LOCK_SH - Shared lock (reader). Allow other processes to access the file

LOCK_EX - Exclusive lock (writer). Prevent other processes from accessing the file

LOCK_UN - Release a shared or exclusive lock

LOCK_NB - Avoids blocking other processes while locking

block is optional parameter.

Example

```
<?php
   $file = fopen("myfile.txt","w+");
   // exclusive lock
   flock($file,LOCK_EX))
   fwrite($file," am Writing this line to file");
   // release lock
   flock($file,LOCK_UN);
   fclose($file);
?>
```

5.11.5 Example

Example 5.11.2 *Create a form containing information sl.no, title of the book, publishers, quantity,price read the data from the form and write onto the file using PHP script.*

Model Question Paper, Mark: 10

Solution :

Step 1 : We will create a input form for the user to enter the data. The HTML code is as follows. (It is the same HTML form which we have discussed in the last section)

HTML Document [input.html]

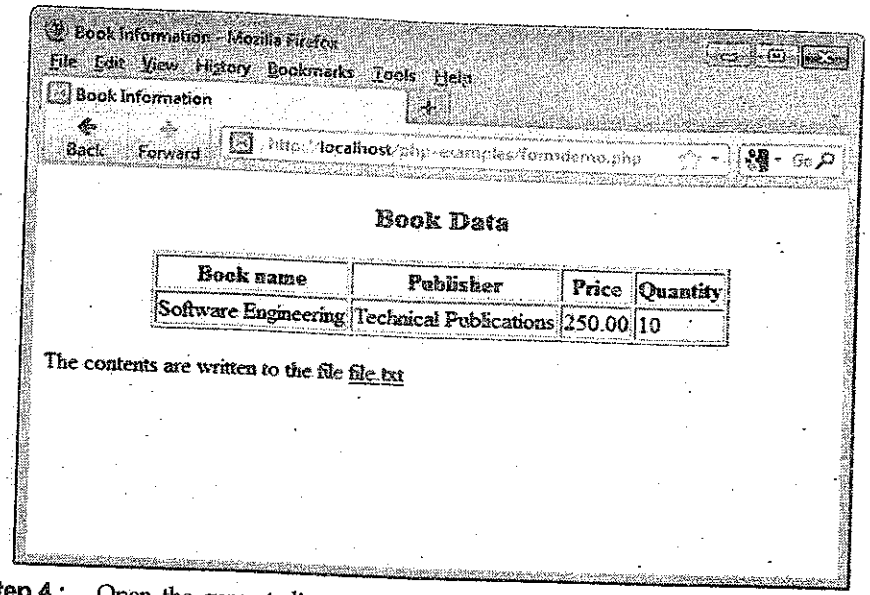
```
<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
<title> Book Order Form </title>
</head>
<body>
<h3> Enter the Book Data </h3>
<form method="post" action="http://localhost/php-examples/formdemo.php">
<table>
<tr>
<td>Sr.No.</td>
<td><input type="text" name="SLNo"></td>
</tr>
<tr>
<td>Book name</td>
<td><input type="text" name="BName"></td>
</tr>
<tr>
<td>Publisher</td>
<td><input type="text" name="PUBName"></td>
</tr>
<tr>
<td>Price</td>
<td><input type="text" name="Price"></td>
</tr>
<tr>
<td>Quantity</td>
<td><input type="text" name="Qty"></td>
</tr>
<tr>
<td><input type="submit" value="Submit"></td>
<td><input type="submit" value="Clear"></td>
</tr>
</table>
</form>
</body>
</html>
```

Step 2 : Following is a simple PHP script which displays the data and then writes it to the file.

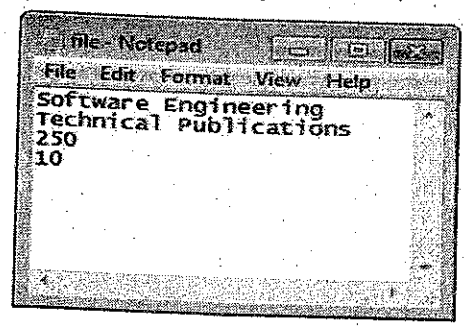
PHP Document

```
<html>
<head>
 <title>Book information</title>
</head>
<body>
<?php
$BName=$_POST["BName"];
$PUBName=$_POST["PUBName"];
$Price=$_POST["Price"];
$Qty=$_POST["Qty"];
?>
<center>
<h3> Book Data </h3>
<table border=1>
<tr>
 <th>Book name</th>
 <th>Publisher</th>
 <th>Price</th>
 <th>Quantity</th>
</tr>
<tr>
 <td><?php print ("$BName"); ?></td>
 <td><?php print ("$PUBName"); ?></td>
 <td><?php printf("%3.2f",$Price); ?></td>
 <td><?php printf("%d",$Qty); ?></td>
</tr>
</table>
</center>
<?php
$my_file = 'file.txt';
$file_handle = fopen($my_file, 'a') or die('Cannot open file: '.$my_file);
fwrite($file_handle, $BName);
fwrite($file_handle, "\r\n");
fwrite($file_handle, $PUBName);
fwrite($file_handle, "\r\n");
fwrite($file_handle, $Price);
fwrite($file_handle, "\r\n");
fwrite($file_handle, $Qty);
fclose($file_handle);
?>
<p>The contents are written to the file <u>file.txt</u></p>
</body>
</html>
```


Step 3 : Now open the web browser and type the address of the html file which you have created in step 1. On clicking the Submit button on the this input form the data will be displayed as follows -



Step 4 : Open the current directory folder(The folder in which your html and PHP scripts are present) . You will obtain a text file file.txt. Just Open the file and see the contents. I have got the following contents -



Review Questions

1. *What is file variable and file pointer? What does an open function returns if it fails ?* May-12, Marks 5
2. *Explain the parameter and action of the fread and write function.* Dec.-12, May-15, Marks 5
3. *Explain locking of file with example* Dec.-13, Marks 5

5.12 Cookies

May 12, 15, Dec 12, 13, 16, Marks 5

In this section we will discuss how to create and read cookies.

5.12.1 Introduction to Cookies

- Cookie is a small file that server embeds in the user's machine.
- Cookies are used to identify the users.
- A cookie consists of a **name** and a **textual value**. A cookie is created by some software system on the server.
- In every HTTP communication between browser and server a header is included. The header stores the information about the message.
- The header part of http contains the cookies.
- There can be one or more cookies in browser and server communication.

5.12.2 PHP Support for Cookies

- PHP can be used to create and retrieve the cookies.
- The cookie can be set in PHP using the function called **setcookie()**
- The syntax for the cookie is -

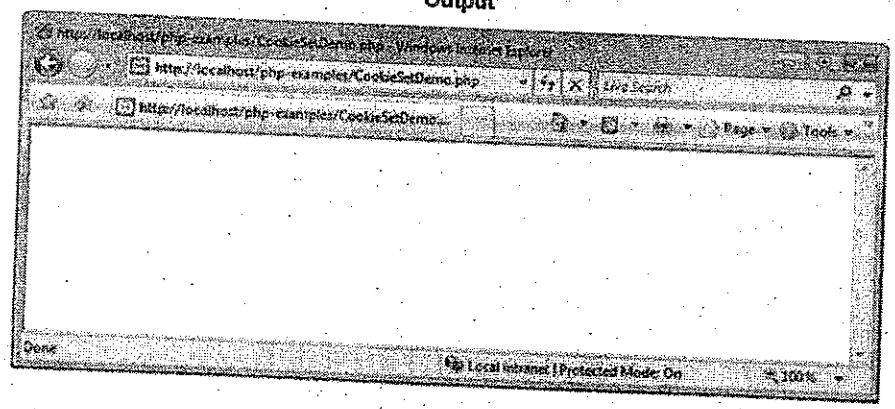
```
setcookie(name,value,expire_period,path,domain)
```

- Following is a simple PHP document which illustrates how to set the cookies -

PHP Document[CookieSetDemo.php]

```
<?php
$Cookie_period=time()+60*60*24*30;
setcookie("Myname", "Monika", $Cookie_period);
?>
```

Output



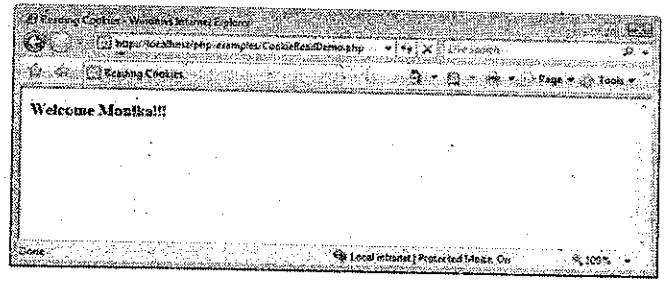
Note that you have got the blank screen it indicates that the cookie is set. In above PHP document we have set the PHP script for one month. Just observer the third parameter of the setcookie function.

Now you can retrieve the cookie and read the value to ensure whether or the cookie is set.

PHP Document[CookieReadDemo.php]

```
<html>
 <head><title>Reading Cookies</title>
<body>
 <?php
if (isset($_COOKIE["Myname"]))
  echo "<h3>Welcome " . $_COOKIE["Myname"]."!!!</h3>";
else
  echo "<h3>Welcome guest!</h3>";
?>
</body>
</html>
```

Output



Program Explanation

The **isset** function is used for checking whether or not the cookie is set. Then using the $_COOKIE the value of the cookie can be retrieved.

Review Questions

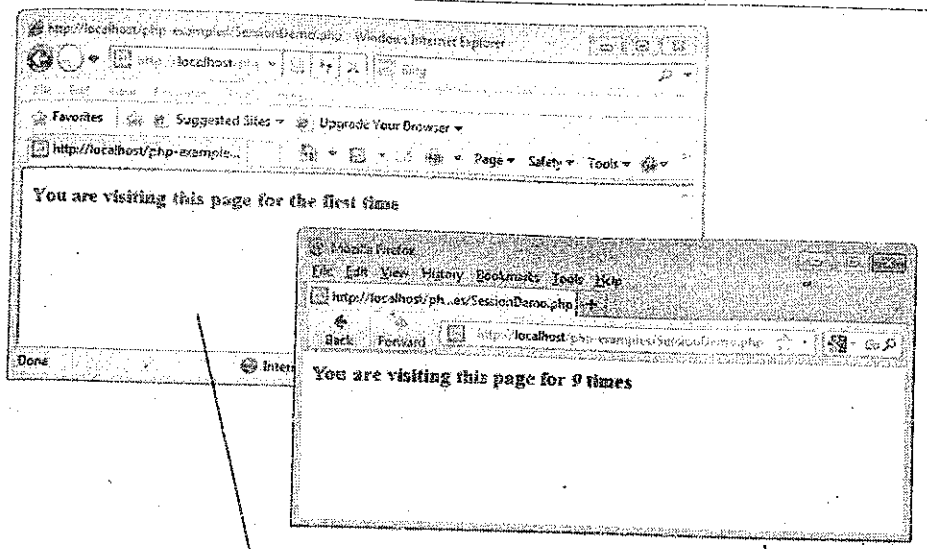
1. *Explain how cookies are extracted using PHP.* Question Bank, May 13, Marks 5
2. *What is cookie ? Explain the parameter and action of the set cookie function.* May 12, Dec 12, Marks 5
3. *Define cookie. Explain how to set and retrieve a cookie.* Dec 13, Marks 5
4. *What is cookie ? Explain how cookies are created using PHP ?* May 15, Marks 5
5. *Write a note on cookies in PHP.* Dec 16, Marks 5

5.15 Session Tracking

- When you open some application, use it for some time and then close it. This entire scenario is named as **session**.
- Sometimes the information about the session is required by the server. This information can be collected during the session. This process is called **session tracking**.
- There exists a session array which often stores the unique session ID for the session.
- PHP keeps track of session by using a function called **session_start()**. Due to the call to **session_start()** function the session ID is created and recorded.
- Following is a simple PHP script in which the information about session is tracked

PHP Document[SessionDemo.php]

```
<?php
session_start();
if(isset($_SESSION['pgvisit']))
{
$_SESSION['pgvisit']=$_SESSION['pgvisit']+1;
echo "<h3>You are visiting this page for <i>" .$_SESSION['pgvisit']."</i> times</h3>";
}
else
{
$_SESSION['pgvisit']=1;
echo "<h3>You are visiting this page for the first time</h3>";
}
?>
```

Open some another web browser say Internet Explorer and type the URL for **SessionDemo.php**. For the first time the above message will be displayed

Open the Mozilla Firefox web browser and type the URL for **SessionDemo.php**. Just refresh this page for many times.

Program Explanation :

In above program, We have started the session by using **session_start()** function.

The **isset()** function checks if the **pgvisit** variable has already been set. If **pgvisit** has been set, we can increment our counter. If **pgvisit** is not set, then we create a set it to 1.

The value of **pgvisit** is displayed on the screen.

Review Question

1. *Write a note on session tracking.* **Question Bank - Marks 5**

Notes

6 Database Access through the Web

Syllabus

Database Access through the Web - Database Access with PHP & MySQL, Potential Problems with Special Characters, Connecting to MySQL & Selecting the Database, Requesting MySQL Operations, A PHP/ MySQL Examples, Database Access with JDBC & MySQL, JDBC & MySQL, Metadata, Examples.

Contents

6.1 Database Access with PHP and MYSQL

Nov. 11, May 12, 13, 14, 16, Dec. 12, 13, 14, 16, Marks 10

- PHP access the database with the help of two HTML documents. The first HTML document is for getting the values from the user and in the second HTML document the PHP code is embedded which processes the user request for accessing the database.
- The first type of HTML is simple as it simply collects the user request. But the second HTML document is for processing the user request for the database.

6.1.1 Potential Problems with Special Characters

- Sometimes there are some special characters in the HTML such as <,> ," and &. To handle these characters from the string there is a function named **htmlspecialchars** which interprets these characters correctly.
- For example : If the string is

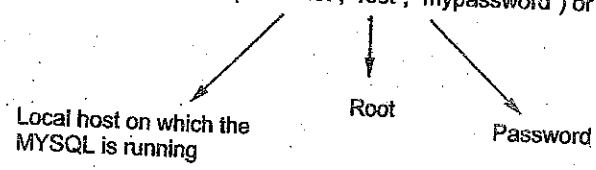
```
$str="Teena & Meena are good friends"
$str=htmlspecialchars($str);
```

- For example the string obtained from the user for the PHP may contain backslashes, single quotes or double quotes. These can cause the problems. To avoid these problems there is one function **magic_quotes_gpc**. This function can be turned on or off. When this function is enabled then all these receiving values will be interpreted correctly. And while displaying these values we can use the function **stripeslashes** function.

6.1.2 Connecting to MYSQL and Selecting the Database

- The PHP function **mysql_connect** connects to the MYSQL server. There are three parameters that can be passed to this function. For example –

```
mysql_connect("localhost", "root", "mypassword") or die(mysql_error());
```



- The database can be selected by using the command **mysql_select_db**. For example

```
mysql_select_db("test")
```

will select the database named **test**.

6.1.3 Requesting MySQL Operations

1. Creating Database

- We can create a database using the function **mysql_query**. The **mysql_error()** function is used to obtain the error messages if any command gets failed.
- mysql_query function in php is used to pass a sql query to mysql database.

Syntax

```
mysql_query (string query [, resource link_identifier])
```

This function returns the query handle for SELECT queries, TRUE/FALSE for other queries, or FALSE on failure.

Example

```
mysql_query("CREATE DATABASE mydb",$con)
```

- The **mysql_connect()** function Open a connection to a MySQL Server.

Syntax

```
mysql_connect ( string server , string username , string password)
```

Returns a MySQL link identifier on success, or FALSE on failure.

Example

```
$conn=mysql_connect("localhost","root","mypassword");
```

- The **mysql_close()** function is used to close the database connection.

Syntax

```
mysql_close (Connection)
```

PHP Example For Creating Database

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
//Create a database
if(mysql_query("CREATE DATABASE mydb",$con))
{
  print "Database created";
}
else
{
  print "Error creating database : " . mysql_error();
```

```
}
mysql_close($conn); //closing the database
?>
```

2. Selecting Database

- The database can be selected using the function **mysql_select_db()**.

Syntax

```
mysql_select_db ( string database_name [, resource link_identifier])
```

Where

mysql_select_db() attempts to select existing database on the server associated with the specified link identifier. It returns TRUE on success, or FALSE on failure.

For example -

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost :3306/mydb ","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
//Select a database
mysql_select_db("mydb", $con);
mysql_close($conn); //closing the database
?>
```

3. Counting Number of Rows

The number of rows present in the database table can be obtained using **mysql_num_rows** function.

Syntax

```
int mysql_num_rows ( resource $result )
```

This returns number of rows in result on success, or NULL on error.

Example

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost :3306/mydb ","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
//Select a database
```



```
mysql_select_db("mydb", $con);
$num_rows = mysql_num_rows($result);
//Print number of rows
echo "Total number of rows are $num_rows";
mysql_close($conn); //closing the database
?>
```

4. Counting Number of Fields

The mysql_num_fields() is used to get number of fields of the table.

Syntax

```
mysql_num_fields(resource_name)
```

It returns the number of fields present in the resource and false on failure

Example

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost:3306/mydb ","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
//Select a database
mysql_select_db("mydb", $con);
$result = mysql_query("select id,name from my_table where id = '1' ");
echo mysql_num_fields($result); // since two fields are fetched, returns 2
?>
```

5. Creating Table

Before creating the table a database must be created and within which the table can be created. Note that before creating a table the desired database must be selected.

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost ","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
//Create a database
if(mysql_query("CREATE DATABASE mydb",$con))
{
print "Database created";
}
else
```

```
{
  print "Error creating database : " . mysql_error();
}
mysql_select_db("mydb",$con); //before creating table select the database
$query="CREATE TABLE my_table
      (
            id INT(4),
            name VARCHAR(20)
      )";
mysql_query($query,$con);//Execution of Query

mysql_close($conn); //closing the database
?>
```

6. Inserting Data in Table

For inserting a data into the table we use the INSERT query. For example

```
$query=" INSERT INTO my_table (id,name) VALUES(1,'SHILPA')";
mysql_query($query,$con);//Execution of Query
```

Here is a PHP script in which insert query is used to insert two records in the table

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("mydb",$con); //select the database
$query=" INSERT INTO my_table (id,name) VALUES(1,'SHILPA')";
mysql_query($query,$con);//Execution of Query
$query=" INSERT INTO my_table (id,name) VALUES(2,'MONIKA')";
mysql_query($query,$con);//Execution of Query

mysql_close($conn); //closing the database
?>
```

Sometimes values that can be inserted in the table can be obtained from some another script and these values might be present in the variables. Insertion of such data can be done using $_POST variables. It is as shown below -

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
```



```
mysql_select_db("mydb",$con); //select the database
$query=" INSERT INTO my_table (id,name)
VALUES('$_POST[MyID]','$_POST[MyName]')";
mysql_query($query,$con);//Execution of Query

mysql_close($conn); //closing the database
?>
```

7. Displaying or Retrieving Records

For displaying the records present in the database table, we use SELECT query. For example

```
//Execution of Query for displaying the data
$result=mysql_query("SELECT * FROM my_table");
```

The above execution returns a result handle. Then The **mysql_fetch_array()** is used to retrieve a row of data as an array from a MySQL result handle.

Purpose of mysql_fetch_array() : The mysql_fetch_array() is used to retrieve a row of data as an array from a MySQL result handle.

Syntax :

```
mysql_fetch_array(result, result_type)
```

PHP Script for Displaying records

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("mydb",$con); //select the database
//Execution of Query for displaying the data
$result=mysql_query("SELECT * FROM my_table");
while($row = mysql_fetch_array($result))
  {
  echo $row['id']." ". $row['name'];      //Each record will be displayed
                                          //line by line
  echo "<br />";
  }
mysql_close($conn); //closing the database
?>
```

8. Finding number of affected rows

The mysql_affected_rows query is used to get number of affected rows in previous MySQL operation such as INSERT, DELETE, UPDATE queries.

Syntax

```
mysql_affected_rows(connection)
```

Example

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("mydb",$con); //select the database
$query=" INSERT INTO my_table (id,name) VALUES(1,'SHILPA')";
mysql_query($query,$con);//Execution of Query
$query=" INSERT INTO my_table (id,name) VALUES(2,'MONIKA')";
mysql_query($query,$con);//Execution of Query
echo "Number of rows affected are :".mysql_affected_rows();
mysql_close($conn); //closing the database
?>
```

6.1.4 PHP/MYSQL Example

Example 6.1.1 *Write a PHP script to create a table, insert records into the table, retrieve records from the table. Assume database "Student" and table "cs_student" with fields name, sem, regno, address.* **Question Bank, May-14, Marks 10**

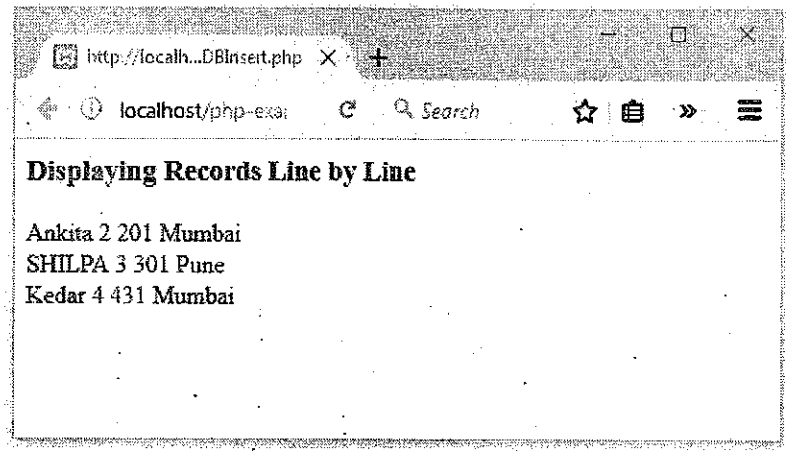
Solution :

```
<?php
// Make a MySQL Connection
$con=mysql_connect("localhost","root","");
if(!$con)
{
die('error in connection'.mysql_error());
}
mysql_select_db("Student",$con); //select the database
$query=" INSERT INTO cs_student(name,sem,regno,address) VALUES ('SHILPA',3,301,
'Pune')";
mysql_query($query,$con);//Execution of Query
$query=" INSERT INTO cs_student(name,sem,regno,address) VALUES ('Kedar', 4,431,
'Mumbai')";
mysql_query($query,$con);//Execution of Query
//Execution of Query for displaying the data
echo "<h3> Displaying Records Line by Line </h3>";
$result=mysql_query("SELECT * FROM cs_student");
while($row = mysql_fetch_array($result))
```



```
{
 echo $row['name'] . " " . $row['sem'] . " " . $row['regno'] . " " . $row['address'] ;
//Each record will be displayed line by line
 echo "<br />";
}
mysql_close($con); //closing the database
?>
```

Output



Example 6.1.2 *Write a PHP script to insert record into the table and retrieve records from the table. Assume a table "my_detail" is already created with fields name, city, phone_no and mail_id.* **Question Bank, Model Question Bank, Nov.-11, Marks 10**

Solution :

```
<?php
// Make a MySQL Connection
$con=mysql_connect("localhost","root","");
if(!$con)
{
die('error in connection'.mysql_error());
}
mysql_select_db("Person",$con); //select the database
$query=" INSERT INTO my_detail(name,city,phone_no,mail_id)
VALUES('Aarti','Pune',2222222222,'aarti123@gmail.com')";
mysql_query($query,$con);//Execution of Query
$query=" INSERT INTO my_detail(name,city,phone_no,mail_id)
VALUES('Soumya','Mumbai',3333333333,'soumya_cool@rediffmail.com')";
mysql_query($query,$con);//Execution of Query
//Execution of Query for displaying the data
```

```
echo "<h3> Displaying Records Line by Line </h3>";
$result=mysql_query("SELECT * FROM my_detail");
while($row = mysql_fetch_array($recult))
 {
 echo $row['name'] . " " .$row['city']." $row['phone_no'] " " .$row['mail_id'] ;
                        //Each record will be displayed
                        //line by line
 echo "<br />";
 }
 mysql_close($con);     //closing the database
?>
```

Example 6.1.3 *Write PHP script to insert records into table and retrieve records from table.*
Assume
fields - fnanie, lname, address, age;
table - tbaddressbook
database - addressbook

Dec. 13, Marks 10

Solution :

```
<?php
// Make a MySQL Connection
$con=mysql_connect("localhost","root","");
if(!$con)
{
die('error in connection' mysql_error());
}
mysql_select_db("addressbook",$con); //select the database
$query=" INSERT INTO tbaddressbook(fname,lname,address,age)
VALUES('AAA','ZZZ','Pune',22)";
mysql_query($query,$con);//Execution of Query
$query=" INSERT INTO tbaddressbook(fname,lname,address,age)
VALUES('BBB','YYY','Mumbai',35)";
mysql_query($query,$con);//Execution of Query
//Execution of Query for displaying the data
echo "<h3> Displaying Records Line by Line </h3>";
$result=mysql_query("SELECT * FROM tbaddressbook");
while($row = mysql_fetch_array($result))
 {
 echo $row['fname'] . " " .$row['lname']." $row['address'] " " .$row['age'] ;
                        //Each record will be displayed
                        //line by line
 echo "<br />";
 }
 mysql_close($con);     //closing the database
?>
```


Review Questions

1. *Explain potential problems associated with special characters.*
   Question Bank Model Question Paper, Marks 5
2. *Write PHP functions used to connect to MYSQL database and selecting a database*
   Question Bank, Marks 5
3. *Construct a PHP script to insert and retrieve records from the database table.*
   Question Bank, May 16, Dec. 16, Marks 10
4. *Illustrate the use of :*
   mysql_query(), mysql_connect(), mysql_select_db(), mysql_num_rows() and mysql_num_fields()
   Question Bank, Dec. 14, 16, Marks 10
5. *Explain the significance of i) mysql-num-rows ii)mysql_affected_rows()* **May 13, Marks 5**
6. *Construct a PHP script to insert records through form into a database.*
   Question Bank Model Question Paper, May 14, Marks 5
7. *Write a PHP script to access the database with MYSQL* **May 12, Dec. 12, Marks 10**
8. *Construct a PHP script to retrieve records from the database table.*
   Question Bank, May 13, Marks 5

6.2 Database Access with JDBC and MYSQL

Nov. 11, May 12, 13, 16, Dec. 12, 16, Marks 5

JDBC is a Java API for database access. A servlet can connect to MYSQL and sends the SQL commands to the database and get the required data. Hence we will first get introduced with structured query language SQL.

6.2.1 Structured Query Language

1. Creating database

```
mysql> CREATE DATABASE mydb;
Query OK, 1 row affected (0.15 sec)
```

2. Displaying all the databases

```
mysql> SHOW DATABASES;
+----------+
| Database |
+----------+
| mydb     |
| mysql    |
| students |
| test     |
+----------+
4 rows in set (0.06 sec)
```

3. Selecting particular database

```
mysql> USE MYDB;
Database changed
```

4. Creating table

We must create a table inside a database hence it is a common practice to use create table command after USE database command. While creating a table we must specify the table fields.

```
mysql> CREATE TABLE my_table(id INT(4),name VARCHAR(20));
Query OK, 0 rows affected (0.28 sec)
```

5. Displaying a table

After creating the table using SHOW command we can see all the existing tables in the current database.

```
mysql> SHOW TABLES;
+-----------------+
| Tables_in_mydb  |
+-----------------+
| my_table        |
+-----------------+
1 row in set (0.00 sec)
```

6. Displaying the table fields

For knowing the various fields of the table we may use following command

```
mysql> DESCRIBE my_table;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| id    | int(4)      | YES  |     | NULL    |       |
| name  | varchar(20) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (0.07 sec)
```

7. Inserting values into the table

We can insert only one complete record at a time. It is as shown below -

```
mysql> INSERT INTO my_table
    -> VALUES(1,'SHILPA');
Query OK, 1 row affected (0.05 sec)
```

8. Displaying the contents of the table

```
mysql> SELECT * FROM my_table;
+----+---------+
| id | name    |
+----+---------+
```



```
|  1 | SHILPA  |
+----+---------+
1 row in set (0.06 sec)
```

We can also write SELECT statement for selecting particular row by specifying some condition such as -

```
mysql> SELECT * FROM my_table where id=1;
or
mysql> SELECT * FROM my_table where name='SHILPA';
```

Thus we can insert the rows into the table by repeatedly giving the INSERT command.

If we want to get the records in sorted manner then we use ORDER BY clause

```
mysql> SELECT * FROM my_table;
+----+---------+
| id | name    |
+----+---------+
|  1 | SHILPA  |
|  2 | SUPRIYA |
|  3 | YOGESH  |
|  4 | MONIKA  |
+----+---------+
4 rows in set (0.00 sec)
mysql> SELECT * FROM my_table ORDER BY name;
+----+---------+
| id | name    |
+----+---------+
|  4 | MONIKA  |
|  1 | SHILPA  |
|  2 | SUPRIYA |
|  3 | YOGESH  |
+----+---------+
4 rows in set (0.00 sec)
```

9. Updating the record

For updating the record in the database following command can be used -

```
mysql > UPDATE my_table
    -> SET name='PRIYANKA'
    -> WHERE id=4;
Query OK, 1 row affected (0.05 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> SELECT * FROM my_table;
+----+---------+
| id | name    |
+----+---------+
```

```
|  1 | SHILPA    |
|  2 | SUPRIYA   |
|  3 | YOGESH    |
|  4 | PRIYANKA  |
+----+-----------+
4 rows in set (0.00 sec)
```

10. Deleting record

For deleting particular record from a database

```
mysql> DELETE FROM my_table
  -> WHERE id=3;
Query OK, 1 row affected (0.04 sec)
```

Then use SELECT statement for displaying the contents of the table we use following command.

```
mysql> SELECT * FROM my_table;
+----+-----------+
| id | name      |
+----+-----------+
|  1 | SHILPA    |
|  2 | SUPRIYA   |
|  4 | PRIYANKA  |
+----+-----------+
3 rows in set (0.00 sec)
```

11. For deleting the table

The table can be deleted using the command

```
mysql>drop table my_table;
```

6.2.2 JDBC Perspectives

Open DataBase Connectivity (ODBC) provides a standard software API method for using databases. We can use ODBC for servlets or JSP by using the bridging configuration because direct ODBC support for Java Servlets/JSP is not available. Hence the bridging configuration is introduced by means of JDBC.

JDBC stands for **Java DataBase Connectivity**. JDBC is nothing but an API (i.e. Application Programming Interface). It consists of various classes, interfaces, exceptions using which Java application can send SQL statements to a database. The SQL is a Structured Query Language used for accessing the database.

JDBC is useful for both application developers and JDBC driver vendors.

The JDBC specification is prepared by Sun Microsystems. Any third party vendor can design their own JDBC drivers using this specification. These JDBC drivers are then used by the application developers for getting connected to the database.

JDBC is specially used for having connectivity with the RDBMS packages (such as Oracle or MYSQL) using corresponding JDBC driver.

6.2.1 How JDBC Works ?

Following is a way by which JDBC works -

1) First of all Java application establishes connection with the data source.
2) Then Java application invokes classes and interfaces from JDBC driver for sending queries to the data source.
3) The JDBC driver connects to corresponding database and retrieves the result.
4) These results are based on SQL statements which are then returned to Java application.
5) Java application then uses the retrieved information for further processing.

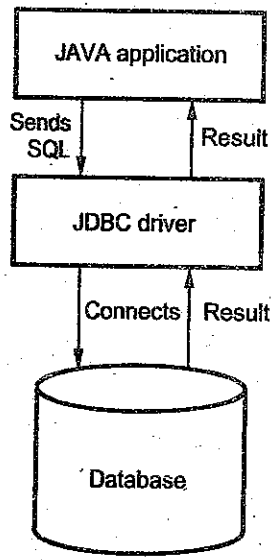


Fig. 6.2.1 Role of JDBC

6.2.2 Difference between JDBC and ODBC

The JDBC is similar to the ODBC i.e. Open Database Connectivity. But the ODBC is used for managing any kind of database and JDBC is specially created to support the Java programs.

Uses of JDBC

1. JDBC helps the client to store and retrieve the data to the databases.
2. JDBC allows the client to update the databases.

Following are the components that are used for establishing the connection with the database using JDBC .

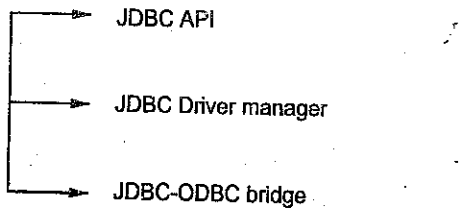


Fig. 6.2.2

The JDBC API classes are supported by the java package **java.sql**. Hence we must import **java.sql.*** in our program.

We have to use following statement for referring the JDBC-ODBC Bridge.

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

There is a JDBC **Driver Manager** which connects your java application or servlet to the JDBC driver. We can try to connect our Java program to my MS Access database with JDBC-ODBC Bridge using the DriverManager.getConnection() method, in which the first parameter is passed as jdbc:odbc:*your_database_name*. The JDBC-ODBC driver helps to translate the JDBC function calls into the ODBC function calls. The JDBC for any database can be implemented only when the ODBC driver is available. Note that JDBC-ODBC bridge is implemented in **sun.jdbc.odbc**.

6.2.3 JDBC Architecture

The JDBC architecture consists of two major components : JDBC API and JDBC driver types.

The JDBC API is a set of classes, interfaces and exceptions used for establishing connection with data source. This JDBC API is defined in the **java.sql** and **javax.sql** packages. We use following core JDBC classes and interfaces that belong to java.sql package.

DriverManager - When Java application needs connection to the database it invokes the DriverManager class. This class then loads JDBC drivers in the memory. The DriverManager also attempts to open a connection with the desired database.

Connection - This is an interface which represents connectivity with the data source. The connection is used for creating the Statement instance.

Statement - This interface is used for representing the SQL statements. Some examples of SQL statements are

```
SELECT * FROM students_table;
UPDATE students_table set name='Neel' where roll_no='1';
DELETE from students_table WHERE roll_no='10';
```

There are two specialized Statement types : PreparedStatement and CallableStatement. The PreparedStatement represents the precompiled SQL statements.

For instance :

```
SELECT * from students_table WHERE name=?
```

The placeholder represented by ? is used in this type of statement. There are special type setter methods that assign the values to the placeholders before the SQL statement is executed.

CallableStatement is represents the stored procedures. These are similar to PreparedStatement. In this type of Statements we can assign the methods for the types of output arguments.

ResultSet - This interface is used to represent the database result set. After using SELECT SQL statement, the information obtained from the database can be displayed using ResultSet.

SQLException - For handling SQL exceptions this interface is used.

6.2.4 JDBC Driver Types

There are four types of JDBC drivers and those are -

Type 1 driver : JDBC-ODBC bridge.

These drivers are used for testing JDBC applications against ODBC data source. These are slowest of all the drivers.

Type 2 driver : Partial Java driver.

These drivers use native database API for data access. These drivers allow the use of wrapper classes.

Type 3 driver : Pure Java driver for accessing middleware server.

This driver is written in Java language only. It first of all accesses the middle level server which in turn invokes the database.

Type 4 driver : Pure Java driver for direct access to database.

This driver is also written in Java language only. It is most commonly used driver and designed for specific vendor's databases. This driver is most efficient among all the other JDBC drivers.

6.2.5 Three-tier Architecture

Since Java supports multi-tier applications many important web applications can be created. A multi-tier architecture is basically a client-server architecture in which the application is to be executed by more than one programs. Typically in multi-tier architecture the middleware is a server to which the client submits some data and then the server sends this input data to the database. Following block diagram shows the multi-tier architecture.

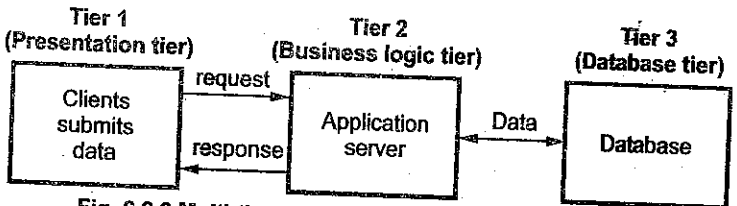


Fig. 6.2.3 Multi-tier architecture (Three tier architecture)

In this multi-tier architecture -

The presentation tier contains a program using which user can submit his request.

The request is then passed to application server which processes it. Sometimes external data may be required application server to process the request.

There is a database at the backend which contains data required by application server.

Finally application server prepares the response object and request is fulfilled by presenting the requested resource to the user.

6.2.6 JDBC and MYSQL

Following steps are used to connect JDBC to MYSQL

Step 1 : Import java.sql.* package in the JDBC program

Following line can be included in your JDBC program at the beginning.

```
import java.sql.*
```

Step 2 : Load JDBC Driver.

The JDBC driver for MYSQL can be loaded using following statement

```
Class.forName("com.mysql.jdbc.Driver");
```

Step 3 : Get connection using the Driver Manager

```
conn = riverManager.getConnection("jdbc:mysql://localhost:3306/my_database","root","password");
```

Step 4 : Create Statement

```
stmt = conn.createStatement();
```

Step 5 : Execute Query

```
String sql = "SELECT Roll,StudName FROM my_table";
```



```
ResultSet rs = stmt.executeQuery(sql);
```

Step 6 : Display the result

Above steps can be illustrated for following example

Example 6.2.1 *Write JDBC program that connects to MYSQL database and display the contents of the database table.*

Solution :

```
import java.sql.*; // Step 1
public class JDBCDemoInsert
{
  public static void main(String[] args)
  {
    Connection conn = null;
    Statement stmt = null;
    try
    {
      Class.forName("com.mysql.jdbc.Driver"); //Step 2
      System.out.println("Connecting to a selected database...");
      //Step 3
      conn = DriverManager.getConnection("jdbc:mysql:
      //localhost:3306/my_database","root","password");
      System.out.println("Connected database successfully...");

      System.out.println("Creating statement...");
      stmt = conn.createStatement();//Step 4
      String sql = "SELECT Roll,StudName FROM my_table";
      ResultSet rs = stmt.executeQuery(sql); //Step 5
      while(rs.next()) //Step 6
      {
        //Retrieve by column name
        int roll = rs.getInt("Roll");
        String name = rs.getString("StudName");
        //Display values
        System.out.print("RollNo: " +roll);
        System.out.println(", StudentName: " +name);
      }
      rs.close();
    }
    catch(SQLException se)
    {
      se.printStackTrace();
    }
    catch(Exception e)
    {
```

```
            e.printStackTrace();
        }
        finally
        {
            try
            {
                if(stmt!=null)
                    conn.close();
            }
            catch(SQLException se)
            {   }
            try{
                if(conn!=null)
                    conn.close();
            }
            catch(SQLException se)
            {
                se.printStackTrace();
            }//end finally try
        }//end try
    }//end main
}//end of class
```

Review Question

1. *Write a note on JDBC and MYSQL* **Dec. 12, Marks 5**

6.2.7 Metadata

Metadata is a data that describes other data. For example ResultSetMetadata describes the result set, such as number of columns in the result set, the data types of those columns, thoer names, nullability and so on.

The resultset interface is used to represent the database result set. After using the SELECT SQL statement, the information obtained from the databse can be displayed using the **ResultSet.** Using the object of **Resultset** we can scroll forward or backward.

The **ResultSetMetaData** interface is used to retireve the number of columns, size of each column in the table, name of the table and so on.

6.2.8 Example

Following is a simple servlet which makes use of the **ResultSetMetaData** for retrieving the information about the fields in the table.

```
import java.io.*;
import java.util.*;
import java.sql.*;
import javax.servlet.*;
```



```
import javax.servlet.http.*;
public class metadataDemo extends HttpServlet
{
Connection conn=null;
ResultSet rs=null;
Statement stmt=null;
public void service(HttpServletRequest request,HttpServletResponse response)throws
IOException,ServletException
{
 PrintWriter out=response.getWriter();
 try
 {
  Class.forName("com.mysql.jdbc.Driver").newInstance();
  conn=DriverManager.getConnection ("jdbc:mysql://localhost/test","root","mypassword");
 }
 catch (ClassNotFoundException e) { e.printStackTrace(); }
 catch (SQLException e) { e.printStackTrace(); }
 catch (Exception e) { e.printStackTrace(); }
   try
   {
response.setContentType("text/html");
out.print("Connected to mysql!!!");
stmt=conn.createStatement();
rs=stmt.executeQuery("select * from myexample");
ResultSetMetaData rsMetaData = rs.getMetaData();
int n = rsMetaData.getColumnCount();
out.print("<br/>"+"Obtaining column information using resultSet MetaData "+"<br/>");
out.print("Total number of columns in the table are "+ n+"<br/>");
for (int i = 1; i <= n; i++)
{
        out.print("column number: <b>" + i+"</b><br/>");
        // indicates the designated column's normal maximum width
        out.print("Size of column:<b>
        "+rsMetaData.getColumnDisplaySize(i)+"</b> <br/>");
        // displays column's name.
        out.print("Name of column:<b>"
            +rsMetaData.getColumnName(i)+"</b><br/>");
        // displays column's table name.
        out.print("Name of parent table:<b>"+
        rsMetaData.getTableName(i)+"</b><br/>");
        // indicates whether the designated column is automatically incrementable
        out.print("Auto increment?:<b>"+rsMetaData.isAutoIncrement(i)+"
        </b><br/>");
}
   }
        catch (SQLException e){}
```

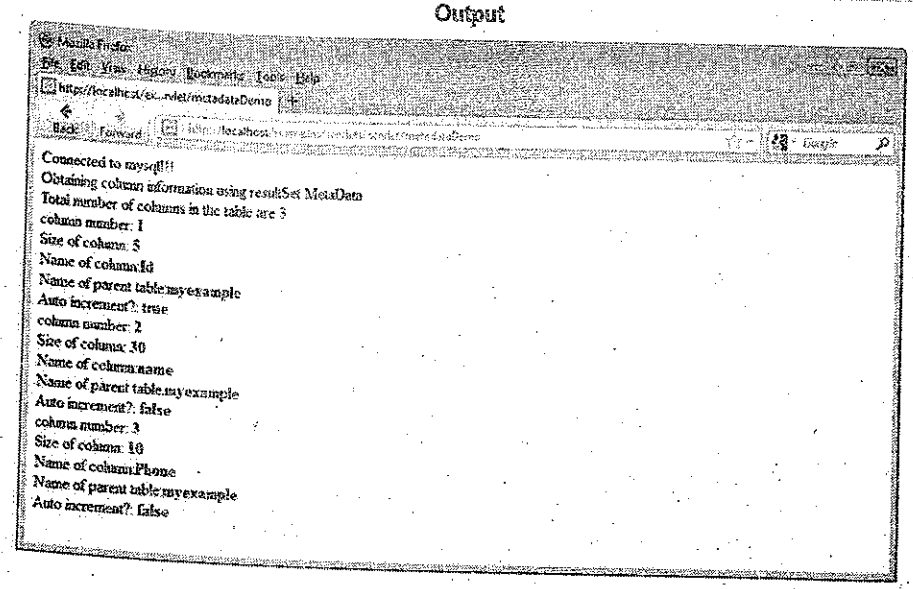
This function first obtains the object for the metadata.

Then using this object the total number of columns in the table can be obtained.

Displaying the information about the table using metadata

```
finally
{
try
{
    if(rs!=null)
{
    rs.close();
    rs=null;
}
if(stmt!=null)
{
    stmt.close();
    stmt=null;
}
if(conn!=null)
{
    conn.close();
    conn=null;
}
}
catch (SQLException e) { }
}
out.close();
}//end of method service
}//end of class
```

Output



Review Questions

1. *Discuss how the collection of metadata extracted from a database.*

   Question Bank, Dec. 16, Marks 5

2. *Explain the use of mysql_query() method used to execute SQL queries with its syntax.*

   Question Bank, Nov. 11, May 13, 16, Marks 5

3. *Explain the steps involved in accessing mySQL database through JDBC.*

   Question Bank, Model Question Paper, Nov. 11, Dec. 16, Marks 10

4. *Create database table to illustrate the insert, delete,update command* May 12, Marks 5

5. *Explain with general form INSERT, DELETE, UPDATE, and DROP SQL commands*

   Dec. 12, Marks 10

❑❑❑

Notes

7 Java Web Software

Syllabus

Introduction to Servlets, Overview, Details, Servlet Containers, The NetBeans IDE, Storing information on Clients, Cookies, Servlet support for Cookies, Examples, JavaServer Pages(JSP), Motivation for JSP, JSP Documents, The Expression Language, The JSTL control action elements, JavaBeans, Model-View-Controller Application Architecture, JavaServer Faces, The tag libraries, JSF event handling, An example application.

Contents

7.1 Introduction to Servlet

7.1.1 Overview

- Servlets are basically the Java programs that run on server. These are the programs that are requested by the XHTML documents and are displayed in the browser window as a response to the request.
- The servlet class is instantiated when web server begins the execution.
- The execution of servlet is managed by **servlet container.**
- The servlet container is used in java for dynamically generate the web pages on the server side. Therefore the servlet container is the part of a web server that interacts with the servlet for handling the dynamic web pages from the client.
- Servlets are most commonly used with HTTP (i.e. Hyper Text Transfer Protocol) hence sometimes servlets are also called as "HTTP Servlet".
- The main purpose of servlets is to add up the functionality to a web server. With the same intension CGI i.e. **Common Gateway Interface** is developed. The CGI is language-independent interface that allows a server to start an external process. But there are some drawbacks of CGI and those are -
  - CGI are less efficient than servlet
  - CGI programs are not always secure
  - It is platform dependent.

Working of How Servlet Works ?

Before learning the actual servlet programming it is very important to understand how servlet works.

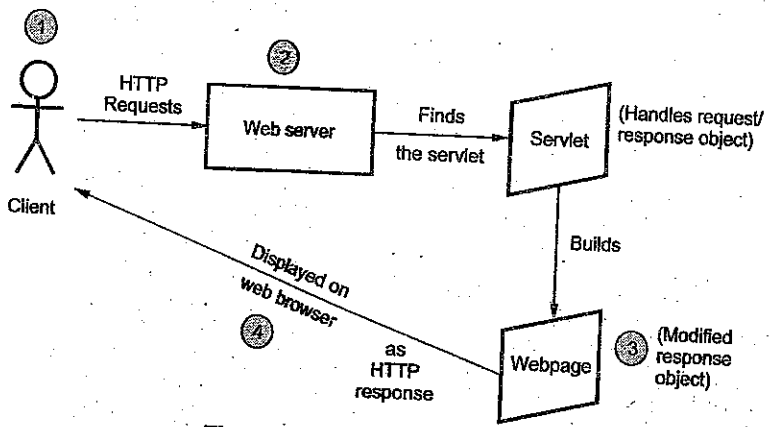


Fig. 7.1.1 How servlet works ?

1. When a client make a request for some servlet, he/she actually uses the **Web browser** in which request is written as a URL.
2. The web browser then sends this request to **Web server**. The web server first finds the requested servlet.
3. The obtained servlet gathers the relevant information in order to satisfy the client's request and builds a **web page** accordingly.
4. This web page is then **displayed** to the client. Thus the request made by the client gets satisfied by the servlets.

- In the life cycle of servlet there are three important methods. These methods are **1. Init 2. Service 3. Destroy**
- The client enters the URL in the web browser and makes a request. The browser then generates the HTTP request and sends it to the Web server. (Refer Fig. 7.1.2)
- Web server maps this request to the corresponding servlet.

1. **Init ( ) Method :** The server basically invokes the **init()** method of servlet. This method is called only when the servlet is loaded in the memory for the first time. Using this method initialization parameters can also be passed to the servlet in order to configure itself.
2. **service ( ) Method :** Server can invoke the service for particular HTTP request using **service()** method. The servlets can then read the data provided by the HTTP request with the help of **service()** method.
3. **destroy ( ) Method :** Finally server unloads the servlet from the memory using the **destroy()** method.

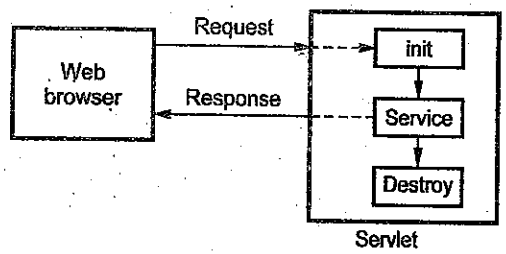


Fig. 7.1.2 Life cycle of servlet

7.2 Details

- When we write a servlet program, it is necessary to - i) either implement **Servlet** interface or ii) extend a class that impements **Servlet** interface.
- While implementing **Servlet** interface we must include **javax.servlet** package.
- **GenericServlet** class is a predefined implementation of **Servlet** interface. Hence we can extend GenericServlet class in our servlet program. Similarly, the

HttpServlet class is a child class of GenericServlet class, hence we can extend this class as well while writing the servlet program.

- Hence following are the two ways by which we can write servlet program

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
public class Test extends GenericServlet
{
 //body of servlet
}
```

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
public class Test extends HttpServlet
{
 //body of servlet
}
```

- The servlet gets the request from the client for some service. The servlet then processes the request and sends the response back to the client. In order to handle these issues **HttpServletRequest** and **HttpServletResponse** are used in servlet program.
- These requests are handled with the help of some methods that are popularly known as **methods of HttpServlet**. These methods are as follows

Method	Purpose
doGet	This method handles the HTTP GET request
doPost	This method handles the HTTP POST request
doPut	This method handles the HTTP Put request
doDelete	This method handles the DELETE request

The doGet and doPost methods

- The **doGet** method requests the data from the source.
- The **doPost** method submits the processed data to the source.
- The protocol of **doGet** method is as follows

Protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException,IOException

  - The **ServletException** and **IOException** are thrown to handle the Servlet problems gracefully.
  - The **HttpServletRequest** request : contain the client request made by client.
  - The **HttpServletResponse** response : contains the response made by servlet back to the client.

- The protocol of doPost method is same as doGet method. It is as follows -

Protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException,IOException

- The GET request is more efficient than the POST request.
- The GET request is less secure than the POST request.

How to Write Servlet Program ?

Open Notepad and write the first servlet code to display Greeting messages. It is as follows

FirstServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class FirstServlet extends HttpServlet
{
 public void doGet(HttpServletRequest request,HttpServletResponse response)
 throws IOException,ServletException
 {
  response.setContentType("text/html");
  PrintWriter out=response.getWriter();
  out.println("<html>");
  out.println("<head>");
  out.println("<title>My First Servlet</title>");
  out.println("<body>");
  out.println("<h1>Hello How are U?</h1>");
  out.println("<h2>I am enjoying this Servlet Application</h2>");
  out.println("<h3>See You later!</h3>");
  out.println("</body>");
  out.println("</html>");
 }
}
```

Program Explanation :

- In the above program, we have imported following files,

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
```

- Out of these files **java.io** package is useful for taking care of I/O operations.
- The **javax.servlet** and **javax.servlet.http** are important packages containing the **classes** and **interfaces** that are required for the operation of servlets. The most commonly used **interface** from **javax.servlet** package is **Servlet**. Similarly most

commonly used class in this package is **GenericServlet**. The **ServletRequest** and **ServletResponse** are another two commonly used interfaces defined in **javax.servlet** package.

- In the **javax.servlet.http** package **HttpServletRequest** and **HttpServletResponse** are two commonly used interfaces. The **HttpServletRequest** enables the servlet to read data from the HTTP request and **HttpServletResponse** enables the servlet to write data to HTTP response. The **cookie** and **HttpServlet** are two commonly used classes that are defined in this package.
- We have given class name **FirstServlet** which should be derived from the class **HttpServlet**. (Sometimes we can derive our class from **GenericServlet**).
- Then we have defined **doGet method** to which the HTTP request and response are passed as parameters. The commonly used basic exceptions for the servlets are **IOException** and **ServletException.**
- The MIME type is specified as using the **setContentType()** method. This method sets the content type for the HTTP response to type. In this method "text/html" is specified as the MIME type. This means that the browser should interpret the contents as the HTML source code.
- Then an output stream is created using **PrintWriter()**. The **getWriter()** method is used for obtaining the output stream. Anything written to this stream is sent to the client as a response. Hence using the object of output stream 'out', we can write the HTML source code in **println** method as HTTP response.

How to Execute Servlet Program ?

Step 1 : Compile the above program using the javac command at command prompt.

```
D:\test>javac FirstServlet.java
```

The class file for this program gets generated.

Step 2 : Before running any servlet program, it is necessary to have

1. JDK installed
2. Tomcat installed.
3. Class path for above two packages must be set.

   For Tomcat installation, I prefer to install the package XAMPP. The XAMP contains a directory for tomcat.

Step 3 : Copy the class file generated in Step 1 to the path

C : \xampp\tomcat\webapps\examples\WEB-INF\classes

Step 4 : Go to the directory

C : \xampp\tomcat\webapps\examples\WEB-INF

Open **web.xml** file and edit it as follows

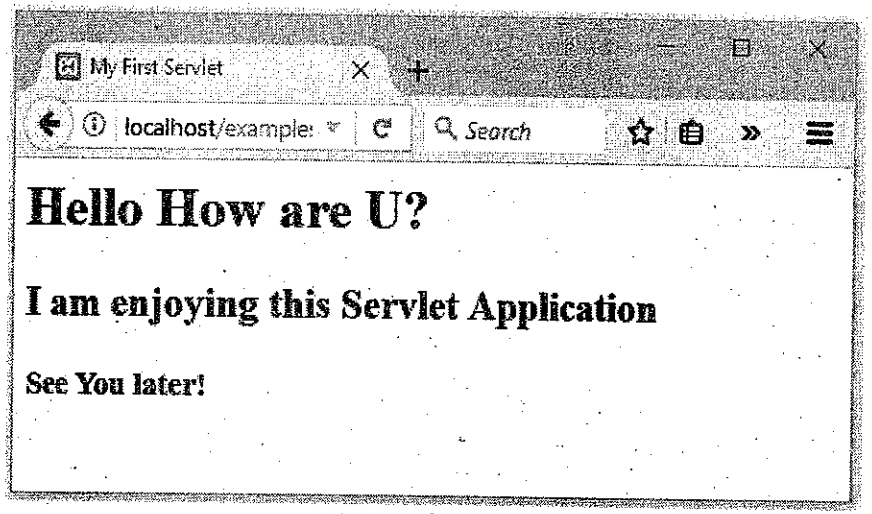
```
<servlet>
    <servlet-name>FirstServlet</servlet-name>
    <servlet-class> FirstServlet </servlet-class>
</servlet>
...
...
<servlet-mapping>
    <servlet-name> FirstServlet </servlet-name>
    <url-pattern>/servlet/ FirstServlet </url-pattern>
</servlet-mapping>
```

Step 5 : Start tomcat and xampp

Step 6 : Open web browser and type the command

http ://localhost/examples/servlet/FirstServlet

The output will be



Example 7.2.1 *Write a web application using servlet to compute an area of a circle. Get the radius from the client. Write necessary web.xml file.*

Solution :

Step 1 : Create an HTML document for client side. Using this HTML document client can enter the radius for computation of area of circle.

```
<html>
 <head>
  <title>Client Form</title>
 </head>
 <body>
  <form action="http://localhost/examples/servlets/servlet/CircleArea" method="get">
   Enter Radius of the Circle : <input type="text" name="Radius"><br />
  <input type="submit" value="Submit" />
  </form>
 </body>
</html>
```

Step 2 : Create the following servlet program in which the value of radius is read and the area of the circle is computed and will be displayed on the web page.

```
import java.io.*;
import java.net.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class CircleArea extends HttpServlet
{
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
 {
  String radius=request.getParameter("Radius");
  int r=Integer.parseInt(radius);
  response.setContentType("text/html;charset=UTF-8");
  PrintWriter out = response.getWriter();
  out.println("<html>");
  out.println("<head>");
  out.println("<title>Servlet Sum</title>");
  out.println("</head>");
  out.println("<body>");

  double a=3.14*r*r;
  out.println("The area of circle is "+a);
  out.println("</body>");
  out.println("</html>");
  out.close();
 }
}
```

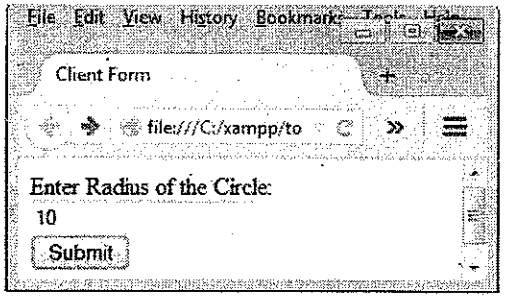
Step 3 : Edit **web.xml** in the WEB-INF folder as follows -

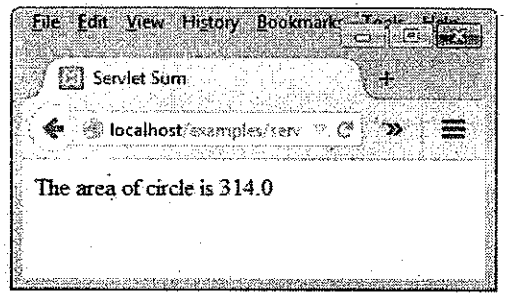
```
...
<servlet>
    <servlet-name>CircleArea</servlet-name>
    <servlet-class>CircleArea</servlet-class>
</servlet>
```



```
 <servlet-mapping>
    <servlet-name>CircleArea</servlet-name>
    <url-pattern>/servlets/servlet/CircleArea</url-pattern>
 </servlet-mapping>
```

Step 4 : The output will be as follows -





Example 7.2.2 *Write a servlet which accept two numbers using POST methods and display the maximum of them.*

Solution : Step 1 : The HTML document for inputting two numbers is as follows -

NumbersInput.html

```
<html>
<head>
<body>
<div align="center">
  <br><br>
  <form action="http://localhost/examples/servlets/servlet/MaxNumber" method="post">
    Enter First Number :
    <input type="text" value="" name="Number1" size="5">
    <br/><br/> Enter Second Number
    <input type="text" value="" name="Number2" size="5">
    <br/><br/><br/>
    <input type="submit" value="Submit">
  </form>
```

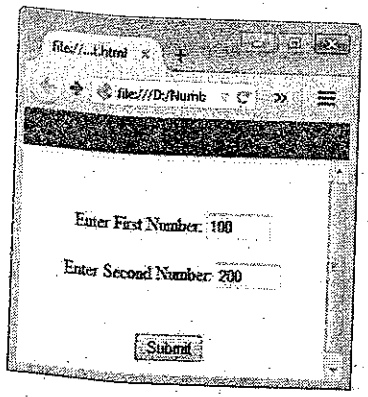
```
</div>
</body>
</html>
```

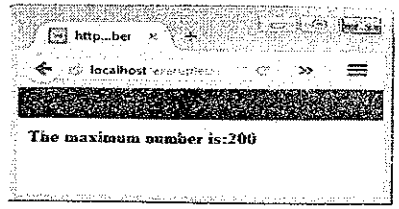
Step 2 : The servlet code that handles the Post method and finds the maximum of the two input numbers is as follows -

MaxNumber.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class MaxNumber extends HttpServlet
 {
  protected void doPost(HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException
  {
   res.setContentType("text/html");
   PrintWriter out=res.getWriter();

   // get request parameters for userID and password
   int a = Integer.parseInt(req.getParameter("Number1"));
   int b = Integer.parseInt(req.getParameter("Number2"));

   if (a>b)
      out.println("<h4>The maximum number is:"+a+"</h4>");
   else
      out.println("<h4>The maximum number is :"+b+"</h4>");

  }
}
```

Step 3 : The output is as follows -





Review Question

1. *Explain doGet and doPost methods of the HttpServlet class.* **Question Bank, Marks 5**

7.3 Servlet Containers

- **Definition :** The servlet container or web container is a component of web server that interacts with Java servlet.
- The servlet container manages the lifecycle activities of servlet.
- Apache Tomcat is an open source web container available under Apache Software License.
- A web container handles requests to servlets, JavaServer Pages (JSP) files, and other types of files that include server-side code.

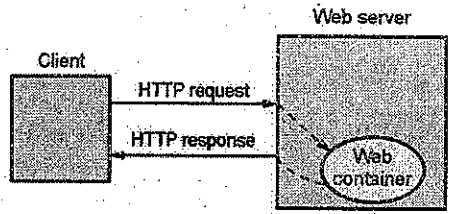


Fig. 7.3.1 Web container

- When client makes a request to web server, the web server handles this request to web container. Web container finds the suitable servlet who can fulfill the request. The job of web container is to send the request to servlet , get it processed and then obtain the response from servlet and back to the client.
  - The container creates two objects – **HttpServletRequest** and **HttpServletResponse**.
  - Then the container creates or allocates a thread for that request and calls the Servlet's **service()** method and passes the request, response objects as arguments.

- The service() method, then decides which servlet method, **doGet()** or **doPost()** to call, based on HTTP Request Method. That means if request is **HTTP GET** then servlet's **doGET** method is used for handling the request.
- Then the Servlet uses response object to write the response back to the client

Review Question

1. *Write a note on Servlet Containers.* **Question Bank, Marks 5**

7.4 The NetBeans IDE

- NetBeans IDE is the official IDE for Java 8.
- NetBeans is an open-source software development project.
- The NetBeans Platform allows applications to be developed from a set of modular software components called modules.
- The NetBeans IDE is primarily intended for development in Java, but also supports other languages, in particular PHP, C/C++ and HTML5.
- NetBeans is cross-platform and runs on Microsoft Windows, Mac OS X, Linux, Solaris and other platforms supporting a compatible JVM.
- **Features of NetBean IDE**
  - **Smart and Fast Code Editing :** This editor supports many languages such as Java, C/C++, XML, HTML,PHP, JSP and so on. The NetBeans Editor indents lines, matches words and brackets, and highlights source code syntactically and semantically.
  - **Efficient Project Management :** It helps to manage the large application in thousands of files and folders. NetBeans IDE provides different views of your data, from multiple project windows to helpful tools for setting up your applications and managing them efficiently.
  - **Cross Platform Support :** NetBeans IDE can be installed on all operating systems that support Java, from Windows to Linux to Mac OS X systems.
  - **Rich set of useful plugins :** The NetBean community is developing rich set of plug-ins for rapid application development and support.

7.5 Storing Information on Clients

- The client and server makes use of HTTP protocol for communication.
- It is called stateless protocol because, neither client not server keeps track of the state of communication session. In this communication each request is independent of the previous request. It cannot remember the earlier

communication between the two. This is really an undesirable condition in any communication.

- But it is possible to store information on client's machine using a technique called **cookies.**

7.5.1 Cookies

Cookies are some little information that can be left on your computer by the other computer when we access an internet.

- Generally this information is left on your computer by some advertising agencies on the internet. Using the information stored in the cookies these advertising agencies can keep track of your internet usage, liking of users.
- For the applications like on-line purchase systems once you enter your personal information such as your name or your e-mail ID then it can be remembered by these systems with the help of cookies.
- Sometimes cookies are very much dangerous because by using information from your local disk some malicious data may be passed to you. So it is upto you how to maintain your own privacy and security.

7.5.2 Servlet Support for Cookies

- The **cookie** class is used to create cookies in servlet.
- The Syntax for constructor for cookies are
  - Cookie()
  - Cookie(String name, String value)
- Various methods used in Cookie are described in following table

S.N.	Method	Purpose
1.	public string getName()	It returns the name of the cookie.
2.	public String getValue()	It returns the value of the cookie.
3.	public string setName()	It sets or changes the name of the cookie.
4.	public String setValue()	It sets or changes the value of the cookie.
5.	public void addCookie(Cookie c)	The cookie is added in the response object of HttpServletResponse interface.
6.	public Cookie[] getCookies()	All the cookies can be returned using this method with the help of HttpServletRequest interface.

- In Servlet following steps are used to support for cookies

Step 1 : Creation of Cookies :

The cookie can be created in Servlet and added to response object using **addCookie** method

```
Cookie cookie=new Cookie("user","XYZ");//creating cookie object
response.addCookie(cookie);//adding cookie in the response
```

Step 2 : Reading Cookies :

The value from the cookie can be obtained using the getName() and getValue() methods.

```
Cookie[ ] my_cookies=req.getCookies();
int n=my_cookies.length;
for(int i=0;i<n;i++)
{
String name=my_cookies[i].getName();
String value=my_cookies[i].getValue();
}
```

7.5.3 Examples

Below is simple HTML form in which a servlet is invoked. This servlet creates a cookie by the name My_Cookie and stores the value entered by you in the textbox of HTML form. You can further get the information stored in the cookie by another servlet program *getCookieServlet*. Hence we will write three programs -

1. Our normal HTML script in which some value is entered in the textbox.
2. The servlet program named mycookieservlet which will set a cookies and take the value entered by you in the HTML form.
3. The another servlet program named *getCookieServlet* which helps us to view the cookie.

HTML Program

```
<html>
<head>
<title>Demo of Cookie</title>
</head>
<body>
<form name="form1" method="post"
action="http://localhost:8080/examples/servlet/mycookieservlet">
<h3> Enter the value for my Cookie: </h3>
<input type=text name="txt_data" size=30 value="">
<input type=submit value="Submit">
</form>
</body>
</html>
```


Servlet Program [mycookieservlet.java]

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class mycookieservlet extends HttpServlet
{
public void doPost(HttpServletRequest req,HttpServletResponse res)
throws ServletException, IOException
{
String txt_data = req.getParameter("txt_data");
// Create cookie
Cookie cookie = new Cookie("My_Cookie",txt_data);
// Adding cookie to HTTP response
res.addCookie(cookie);
// Write friendly output to browser
res.setContentType("text/html");
PrintWriter out = res.getWriter();
out.println("<h2>MyCookie has been set to : ");
out.println(txt_data);
out.println("<br><br><br>");
out.println("This page shows that the cookie has been added");
out.close();
}
}
```

We have first created an object of Cookie class using which the cookie can be added using addCookie() method.

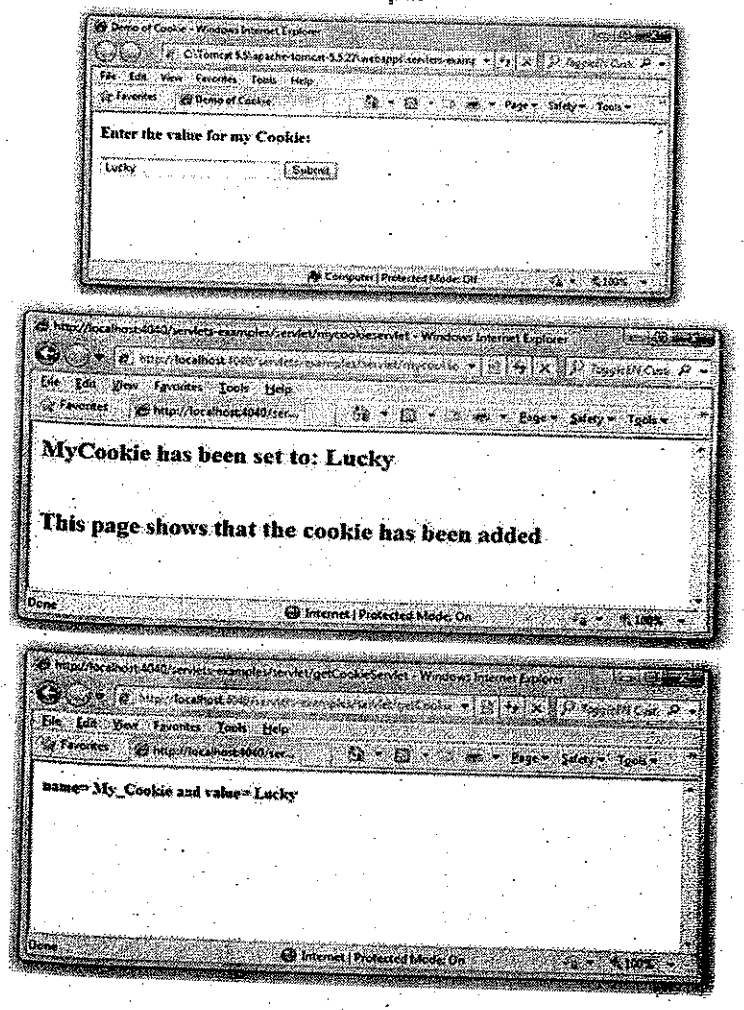
Servlet Program[getcookieservlet.java]

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class getCookieServlet extends HttpServlet
{
public void doGet(HttpServletRequest req,HttpServletResponse res)
throws ServletException,IOException
{
Cookie[ ] my_cookies=req.getCookies();
res.setContentType("text/html");
PrintWriter out=res.getWriter();
out.println("<b>");
int n=my_cookies.length;
for(int i=0;i<n;i++)
{
String name=my_cookies[i].getName();
String value=my_cookies[i].getValue();
```

```
out.println("name= "+name);
out.println("and value= "+value);
}
out.close();
}
}
```

In the above program using **getName()** and **getValue()** functions we can get the name of the cookie as well as the value of the cookie respectively. The output of all the above given three programs is as given below -

Output



Review Questions

1. *Write a note on cookies.*	Question Bank, Marks 5
2. *Explain the different methods of cookies with an example.*	Question Bank, Marks 10

7.6 Java Server Pages (JSP)

- JSP stands for Java Server Pages.
- It is an alternative way than the servlet to build the dynamic web pages. It is built on top of the servlet.

7.6.1 Motivation for JSP

There are several **problems associated with servlets** and those are -

1. For developing a servlet based application, knowledge of Java as well as HTML code is necessary.
2. While developing any web based application, if a look and feel of web based application needs to be changed then the entire code needs to be changed and recompiled.
3. There are some **web page development tools** available using which the developer can develop web based applications. But the servlets do not support such tools. Even if such tools are used for a servlet, we need to change embedded HTML code manually, which is a time consuming, error prone and complicated process.

Difference between Servlet and JSP

Sr. No.	JSP	Servlet
1.	JSP is a scripting language that generates dynamic content	Servlets are the Java programs that can be compiled to generate dynamic content
2.	JSP runs slower than servlet	Servlet runs faster than JSP
3.	In Model View Controller JSP acts as view	In Model View Controller servlet acts as controller
4.	JSP can build custom tags	There is no facility of creating custom tags
5.	JSP can directly call Java beans	Servlets have no facility of calling Java Bean
6.	It is easier to code in JSP	The servlets are basically complex Java programs

7.6.2 JSP Document

Every JSP document contains four elements -

1. **Template Text :** The XML of HTML code can be directly used in JSP document. This is called the static or fixed part of the document. This markup is called **template text.** This part is not modified by JSP container.

2. **Action Elements :** The action elements are used to create the documents dynamically. The action elements is written in the form i) Opening tag ii) Action body iii) Closing tag. For example

<jsp:include>	This tag works as a subroutine. It includes the response from servlet or JSP when request is being processed.
<jsp:forward>	This tag helps in forwarding the current request to servlet or to JSP page.
<jsp:element>	This tag generated the XML elements dynamically.
<jsp:text>	This tag is used to handle template text. When JSP pages are written as XML documents then this tag is used.

3. **Directives :** JSP directives control the processing of entire JSP Page. It gives direction to the server regarding the processing of a page. The directive elements are enclosed within the <%@ .... %> delimiters. For example -

```
<%@page import="java.io.*" %>
<%@page language="java" %>
<%@page contentType="text/html" %>
```

4. **Scriptlets :** Scriptlets are nothing but java code enclosed within <% and %> tags. For example we can write following statement in JSP document

```
<p>
    <% out.print("JSP is eqaul to HTML and JAVA "); %> //Scriptlet
</p>
```

Example of JSP Document

```
<%@ page language="java" contentType="text/html" %>
    <html>
        <head>
            <title>Demo for Template Text</title>
        </head>
        <body bgcolor="gray">
            <h1>Twinkle Twinkle little stars</h1>
            <h2>How I wonder what you are!!!</h2>
            <li>like a diamond in the sky</li>
            <p>
                <% out.print("JSP is eqaul to HTML and JAVA "); %>
            </p>
    </body>
</html>
```

Directive

Template text

Scriptlets

7.6.3 JSP Documents Processing

JSP pages can be processed using **JSP container** only. Following are the steps that need to be followed while processing the request for JSP page -

I. Client makes a request for required JSP page to the server. The server must have JSP container so that JSP request can be processed. For instance : Let the client makes a request for *xyz.jsp* page.

II. On receiving this request the JSP container searches and then reads the desired JSP page. Then this JSP page is straightaway converted to corresponding **servlet.** Basically any JSP page is a combination of template text and JSP **element.** Every **template text** is translated into corresponding **println** statement.

For instance :

```
<html>
<head>
<title>Demo</title>
...
```

```
out.println("<html>");
out.println("<head>");
out.println("<title>Demo</title>");
...
```

Every **JSP element** is converted into corresponding **Java code.** This phase is called **translation phase.** The output of translation phase is a servlet. For instance : our *xyz.jsp* gets converted to *xyzServlet.java.*

III. This servlet is then compiled to generate the servlet class file. Using this class the response can be generated. This phase is called **request processing phase.**

IV. The JSP container thus executes the servlet class file.

V. A requested page is then returned to the client as a **response.** Refer Fig. 7.6.1.

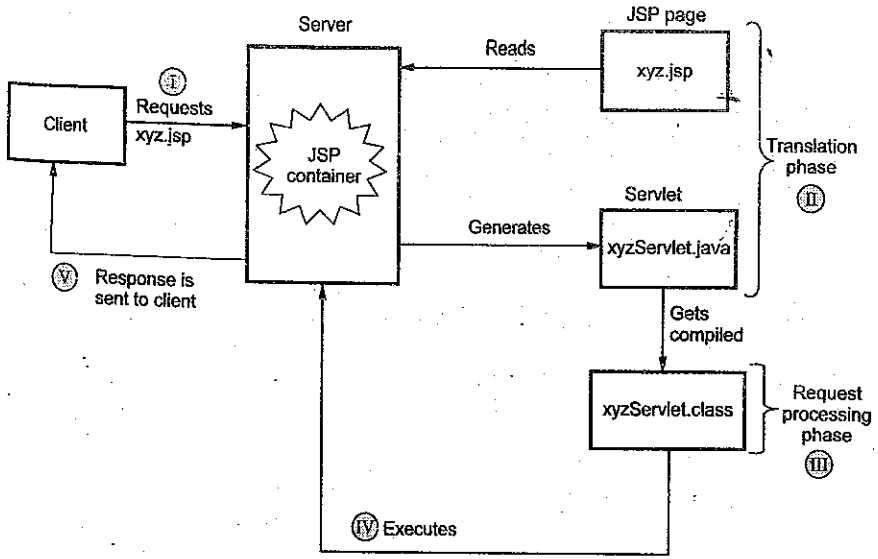


Fig. 7.6.1 JSP processing

Review Questions

1. *Explain three elements associated with JSP.* **Question Bank, Marks 5**
2. *Explain the processing flow of JSP documents with a neat diagram.* **Question Bank, Marks 10**

7.7 How to Write and Execute JSP Document?

For execution of JSP pages following installations are needed.

1. **Java Development Kit :** Download and install the latest version of JDK.
2. **Tomcat WebServer :** The Tomcat is a web container provided by Apache Web server. Now a days XAMPP software package is installed for running JSP, Servlets and PHP programs.

Note that before running any JSP page it is essential to start the tomcat by double clicking the startup.bat file present in the tomcat directory.

Example *Write a simple JSP page for displaying the message "This is my first JSP page!!!"*

We follow these steps to get the JSP page displayed -

Step 1 : First of all open some text editor like Notepad and type the following code.

hello.jsp

```
<%@ page language="java" contentType="text/html" %>
```

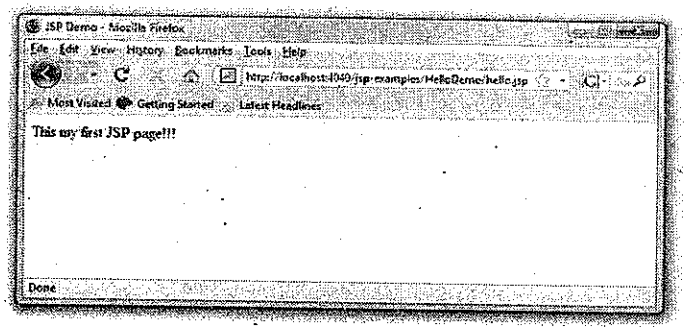


```
<%@ page import ="java.util.*" %>
<html>
<!-- This is basic JSP page -->
<title>JSP Demo </title>
<body>
<%-- Displaying the message on the browser --%>
        <% out.println("This my first JSP page!!!"); %>
    </body>
</html>
```

Step 2 : Copy this file to webapps directory of your tomcat folder.

Step 3 : Start the Tomcat Web server.

Step 4 : Open some suitable web browser like Firefox or Internet Explorer. Type the path for the JSP document and the output will be as follows -



Code explanation

The first line is the **page directive** statement

```
<%@ page language="java" contentType="text/html" %>
```

using this line we are specifying that the language used in this script is Java and the contentType specifies the MIME type which is **text/html**.

Then on the next line we have imported the Java library package **java.util** using import statement -

```
<%@ page import ="java.util.*" %>
```

We can specify the JSP comments in two ways

```
<!-- HTML style comment statement -->
<%-- JSP comment statement --%>
```

As JSP is commonly used along with HTML, it also supports the HTML comments.

For displaying the message on the web browser we have written the message using the Java statement

```
out.println("This my first JSP page!!!");
```

Normally any Java code must be written within <% and %>

Then all the corresponding tags are closed.

7.8 Expression Language

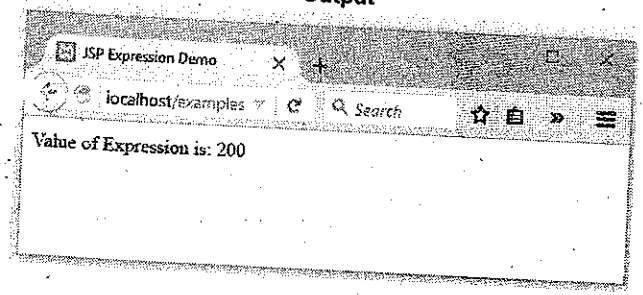
The expression tag is used to represent the expression in JSP page. The syntax of writing expression is -

```
<%= Java Expression %>
```

For example -

```
<html>
  <head>
    <title> JSP Expression Demo</title>
  </head>

<body>
  Value of Expression is :
    <%= (10*20) %>
</body>
</html>
```

Output



7.9 JSTL Control Action Elements

- JSTL stands for JSP Standard Tag Library.
- The JSTL makes use of JSP along with a **tag library**. This tag library is useful for performing some common tasks such as condition execution, loop execution, data procession and so on.
- JSTL allows the programmer to embed the logic in JSP page without using JAVA code. The tag library makes use of some standard set of tags.
- There are five types of tags used in JSTL. These are

Tag	Purpose	uri, prefix and example
Core tag	It is used for performing basic input-output, looping operations, evaluating expressions and so on.	• The url for the core tag is http://java.sun.com/jsp/jstl/core. • The prefix of core tag is c. **Example :** <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
Function tag	This tag is used to support string manipulation and string length functions.	• The url for the functions tags is http://java.sun.com/jsp/jstl/functions • The prefix is fn. **Example :** <%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %>
XML tag	The xml tag is used to for flow control and transformations.	• The url for the xml tags is http://java.sun.com/jsp/jstl/xml • The prefix is x. **Example :** <%@ taglib prefix="x" uri="http://java.sun.com/jsp/jstl/xml" %>
SQL tag	For supporting sql statements this tag is used.	• The url for the sql tags is http://java.sun.com/jsp/jstl/sql • The prefix is sql. **Example :** <%@ taglib prefix="sql" uri="http://java.sun.com/jsp/jstl/sql" %>
Formatting tag	For message formatting, date formatting this tag is useful	• The url for the xml tags is http://java.sun.com/jsp/jstl/fmt • The prefix is fmt. **Example :** <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>

- The **core tag** is used nearly in all the web applications. This tag is useful for performing basic input and output, for looping operation or for evaluating expressions. Various tags used in core library are -

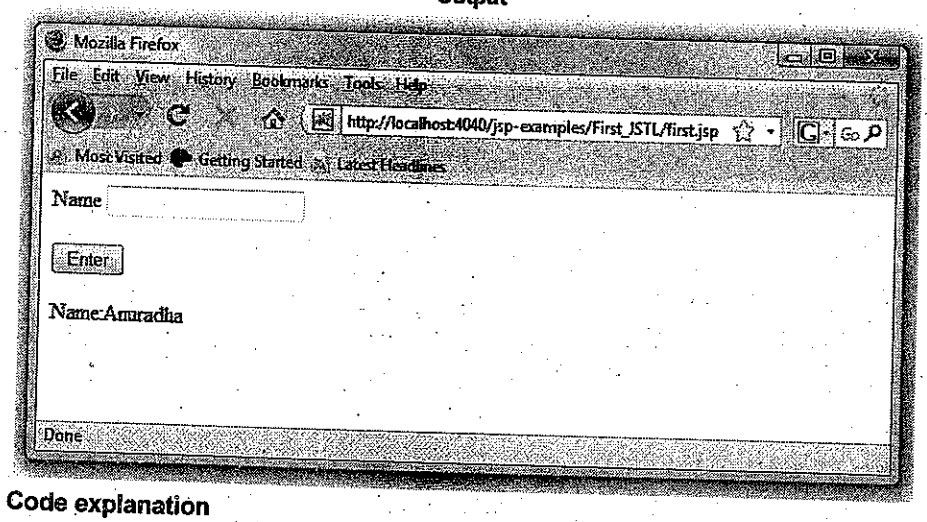
1. <c:out>
2. <c:set>

3. <c:if>
4. <c:choose>
5. <c:when>
6. <c:otherwise>
7. <c:forEach>
8. <c:forTokens>
9. <c:url>
10. <c:redirect>
11. <c:param>
12. <c:import>

Example

```
<%@ page contentType="text/html" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<html>
<body>
<form method=post action="first.jsp">
Name <input type=text name="text1" value=""><br><br>
<input type=submit value="Enter">
</form>
Name:<c:out value="${param.text1}"/><br>
</body>
</html>
```

Output



Code explanation

In above written JSP code we have used **taglib** directive with prefix **c**. Note that we must specify the **uri="http://java.sun.com/jsp/jstl/core"** for every core tag and prefix **"c"**.

We have used **<c:out** for outputting the value written in the text box of the form.

Review Questions

1. *List the five parts of JSTL.*	Question Bank - Marks 5
2. *Illustrate JSTL control action elements with an example for each.*	Question Bank - Marks 10

7.10 JavaBeans

Definition : Javabean is a specialized Java class that can be used as reusable software component.

Characteristics of JavaBeans

1. It contains public no argument constructor.
2. It should be serializable.
3. It provides methods to get and set the values to data attributes. These are called getter and setter method.

Properties

A JavaBean property is a named attribute. These properties are enlisted in the following table.

Method	Description
getPropertyName()	For example, if property name is rollno, then it can be written as getRollno(). This method is called accessor. It returns the value
setPropertyName()	For example, if property name is rollno, then it can be written as setRollno() to. This method is called mutator. It sets the value

Example of Java Bean

We will create JavaBean for entity **Student**.

Step 1 : Create a Java bean program as follows -

Student.java

```
public class Student implements java.io.Serializable
{
   private int roll;
   private String name;
   public Student()
   {
   }
   public void setRoll(int roll)
```

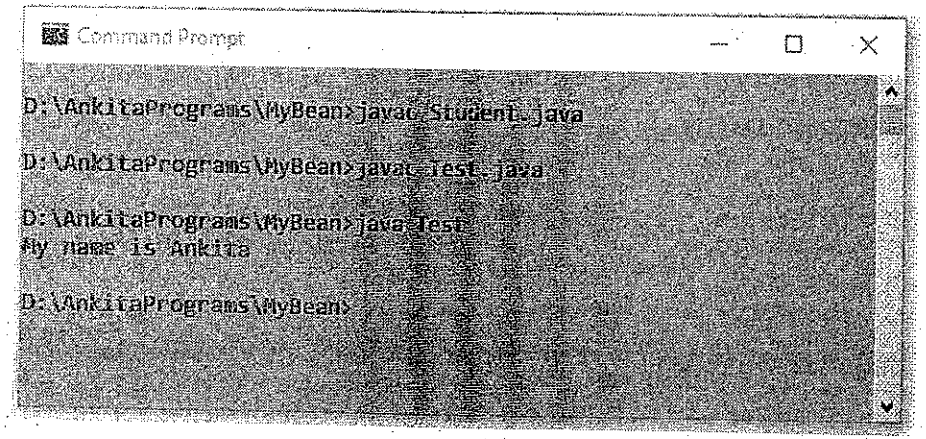
```
    {
        this.roll=roll;
    }
    public int getRoll()
    {
        return roll;
    }
    public void setName(String name)
    {
        this.name=name;
    }
    public String getName()
    {
        return name;
    }
}
```

Step 2 : Now create a driver program that uses the above created **Student** bean.

Test.java

```
public class Test
{
    public static void main(String args[])
    {
        Student st=new Student();
        st.setName("Ankita");
        System.out.println("My name is "+st.getName());
    }
}
```

Step 3 : Now compile the above programs on command prompt window and get the output as follows



7.11 Model View Controller Application Architecture

- Model View Controller or MVC is a software design architectural pattern used for developing web applications.
- The basic idea in MVC design **model** is to **separate out design logic into three parts** - modelling, viewing and controlling.
- Any server application is classified in three parts such as **business logic, presentation** and **request processing.**
- The **business logic** means the coding logic applied for manipulation of application data.
- The **presentation** refers to the code written for look and feel of the web page. For example : background color, font style, font size, placing of controls such as text boxes, command buttons and so on.
- The **request processing** is nothing but a combination of business logic and presentation. The request processing is always done in order to generate the response.
- According to the MVC design model,
  - The **Model** corresponds to business logic,
  - **View** corresponds to presentation and
  - **Controller** corresponds to request processing.
- For example - If user is using the horizontal scrollbar, the

1. The current position of Thumb, its minimum and maximum values are stored and updated by **model.**
2. The **view** module is displays the current status of the scrollbar thumb position. This data is obtained from the **model** module.
3. The **controller** is responsible for accepting the mouse click events, drag of thumb event, when user handles the scroll bar. The controller always communicate with the model for updating the current status of the scrollbar.

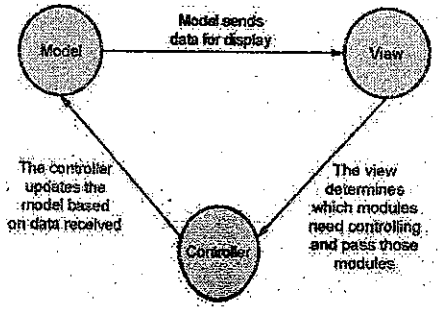


Fig. 7.11.1 MVC architecture

Advantages of MVC Architecture

1. Due to separation of business logic, request processing and presentation activities, it is possible to perform development of all these three simultaneously.
2. Due to separation of responsibilities, the modification or changes can be easily incorporated.
3. Due to separation, finding errors from the modules and correcting them becomes easy and does not affect the working of other module.
4. A model can have multiple views.

Review Question

1. *Write a note on MVC application architecture.* **Question Bank, Marks 5**

7.12 Java Server Faces

- Java Server Faces (JSF) is a web application framework used to develop the web applications with rich user interface.
- This Framework is based on **Model View Controller (MVC)** architecture. There are other technologies that make use of MVC pattern are JSP, Spring, ASP.NET and so on.
- It simplifies the construction of User Interface (UI) components. These UI components can establish communication with databases or event handlers.
- In JSF, the UI components can be created with the help of Application Programming Interface (API).

Advantages of JSF

1. It is component based web framework. Hence the web application can be developed quickly.
2. It allows to create reusable components. This increases the productivity and consistency.
3. It allows implementation of custom components.
4. It has great support for EL expression that improves the user interface code readability.
5. It has inbuilt Template Layout Feature.
6. Validation and Converter implementation is easy.
7. Most of the things in JSF can be done using annotations (@)
8. JSF primeface has good pre-built component suite and it has component.

Disadvantages of JSF

1. This framework is not suitable for high performance applications.
2. It allows very little control over generated HTML/CSS or JavaScript pages.

7.12.1 JSF Architecture

- User submits the request to the application server using web browser. This request is received by JSF Faces Servlet. This servlet is a part of JSF library and it need not be written. It basically acts as a **controller.** It routes the request to appropriate page. It can read config.xml file or managed bean.
- The managed bean contains the form data which can access the backend data or database to retrieve desired data for processing the request. The managed bean acts as a **Model.**
- The Faces Servlet will determine and route the request to appropriate page for display information. Typically these web pages are in the xhtml files. This part acts as a **view**
- Finally the web page is rendered and sent to the web browser of the user.
- Following Fig. 7.12.1 represents this working scenario.

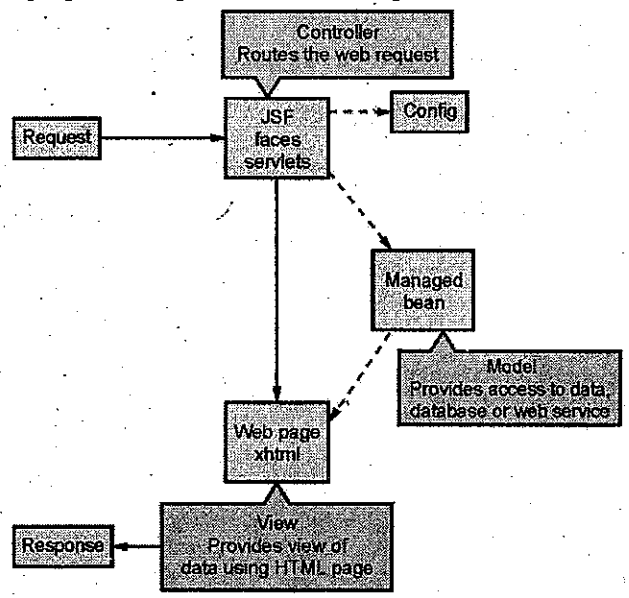


Fig. 7.12.1 JSF architecture

7.13 Tag Libraries

- To develop an JSF application in an convenient way Sun Microsystems, Inc provided some tag libraries. These tags fulfil the basic requirement or functionality while developing the JSF applications.
- Each JSF Tag Libraries have their own namespace, or prefix these are as follows

JSF Component	Tag used	JSF Namespace
Core component	f	http://xmlns.jcp.org/jsf/core
HTML component	h	http://xmlns.jcp.org/jsf/html
Facelets components	ui	http://xmlns.jcp.org/jsf/facelets
Composite components	cc	http://xmlns.jcp.org/jsf/composite

- The Core and HTML tag libraries are the commonly used standard libraries.

1. Core Tag Library

The core library contains the tags that are independent of HTML rendering.

The Standard Syntax to use the JSF core tag is as follows :

```
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:f="http://java.sun.com/jsf/core" >
```

List of some JSF core tags are -

f:attribute	Sets an attribute (key/value) in its parent component.
f:param	Adds a parameter child component to its parent component.
f:event	Adds a component system event listener.
f:converter	Adds an arbitrary converter to a component.
f:convertDateTime	Adds a datetime converter to a component.
f:convertNumber	Adds a number converter to a component.
f:validator	Adds a validator to a component.

2. HTML Tag Library

JSF provides a standard HTML tag library. These tags get rendered into corresponding html output.

The Standard Syntax to use the JSF core tag is as follows :

```
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:h="http://java.sun.com/jsf/html">
```


List of some JSF HTML tags are -

S.N.	JSF UI Component	Description
1.	h:form	HTML form
2.	h:inputText	Single line text input con
3.	h:inputTextArea	Multiline text input control
4.	h:inputSecret	Password input control
5.	h:outputLabel	Label for another component for accessibility
6.	h:outputLink	HTML anchor tag
7.	h:outputText	Single line text output
8.	h:commandButton	Push Button for Submit or Reset
9.	h:selectOneListBox	Single select listbox
10.	h:selectOneMenu	Single select menu
11.	h:selectOneRadio	Radio buttons
12.	h:selectBooleanCheckbox	Checkbox
13.	h:selectManyCheckbox	Set of check boxes
14.	h:selectManyListbox	Multiselect listbox
15.	h:selectManyMenu	Multiselect menu

Review Question

1. *Explain the two standard tag libraries of JSF.* **Question Bank, Marks 5**

7.14 JSF Event Handling

- In JSF event handling can be done using **Event handlers** such as valueChangeListener, actionListener and Application Events. These are described as follows -

S.N.	Event Handler	Purpose
1.	actionListner	When user clicks button or link, the action event gets fired.
2.	valueChangeListner	When user makes changes in input components this value change event gets fired.
3.	Application Events	During JSF lifecycle, these events are fired.

- The form element such as **commandbutton** can be associated with **Listner** method say **processMyAction()** and when this button is clicked the action listener method is invoked.

- The syntax of processMyAction method is

```
public void processMyAction(ActionEvent event) {
//action to be performed to process the event
}
```

7.4 An Example Application

Following is a simple event handling program in which implements handles the event using actionListener

Step 1 : Create and xhtml file on which command button is places. On clicking this button the action listener method is invoked.

test.xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
  xmlns:h="http://java.sun.com/jsf/html"
  xmlns:f="http://java.sun.com/jsf/core">
  <h:head>
    <title>EVENT DEMO</title>
  </h:head>
  <h:body>
  <f:view>
    <h:form>
  <h3>Click Me:</h3>
    <h:commandButton id="myid" value="#{myBean.status}"
    immediate="true" actionListener="#{myBean.processMyAction}"/>
  </h:form>
  </f:view>
    </h:body>
</html>
```

Step 2 : Create a bean (kind of Java program) which will perform the desired action inside the methods

MyBean.java

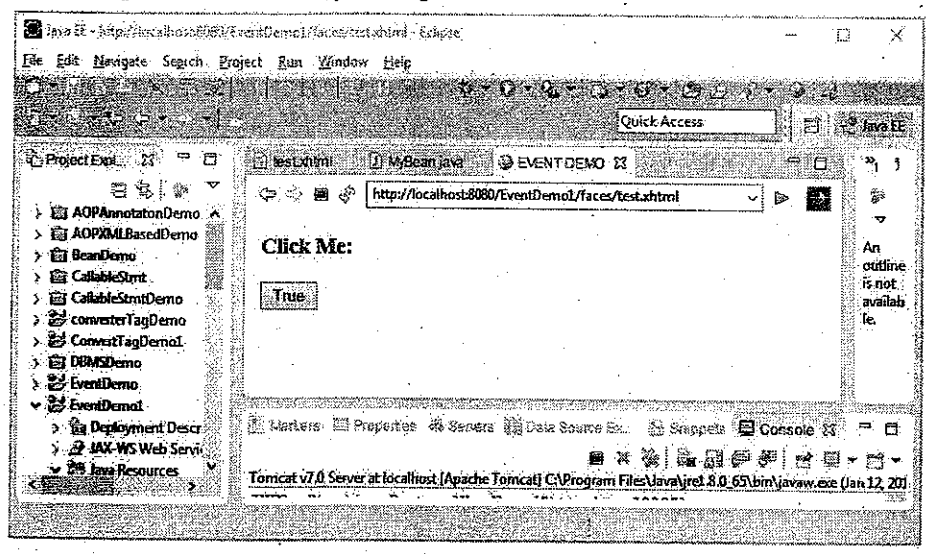
```
package com.mypackage;

import javax.faces.event.ActionEvent;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;

@ManagedBean
@SessionScoped

public class MyBean {
```



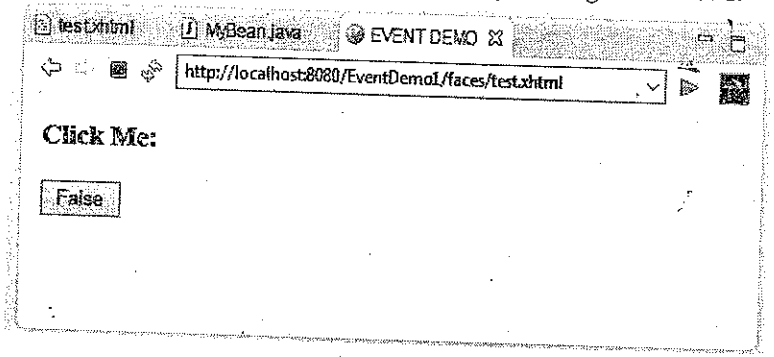
```
    String status = "False";

    public String getStatus() {
    return status;
    }

    public void setStatus(String status) {
    this.status = status;
    }

    /* Action listener method */
    public void processMyAction(ActionEvent event) {

    if (status.equals("False"))
    {
     status = "True";
    }
    else
    {
     status = "False";
    }
  }
}
```

Step 3 : For obtaining the output, Right click on xhml file and run it on server.

The output can be verified by clicking the button on the form



Click the command button and note that the value gets changed as follows.



❑❑❑

Web Programming Lab

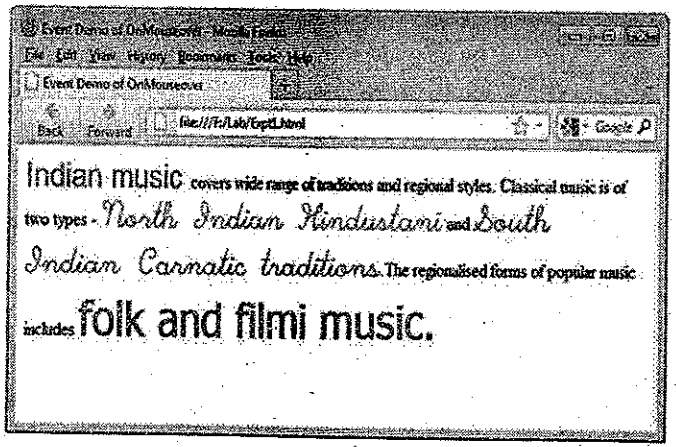
Contents

Experiment 1 : The document must have a paragraph of text that describes your home. Choose atleast three different phrases (3 to 6 words) of this paragraph and make them change font, font style, color and font size when the mouse cursor is placed over them. Each of the different phrases must change to different fonts, font styles, colors and font sizes.

Solution :

```
<!DOCTYPE html>
<html>
<head>
 <title> Event Demo of OnMouseover</title>
</head>
<body>
 <a onmouseover="this.style.color='green';this.style.font='20pt Arial';"> Indian music
</a>covers wide range of traditions and regional styles. Classical music is of two types -
<a onmouseover="this.style.color='red';this.style.font='25pt script';"> North Indian
Hindustani</a> and <a onmouseover="this.style.color='blue';this.style.font='25pt
script';">
South Indian Carnatic traditions</a>.The regionalised forms of popular music includes <a
onmouseover="this.style.color='maroon';this.style.font='30pt Tahoma';">folk and filmi
music.</a>
</body>
</html>
```

Output



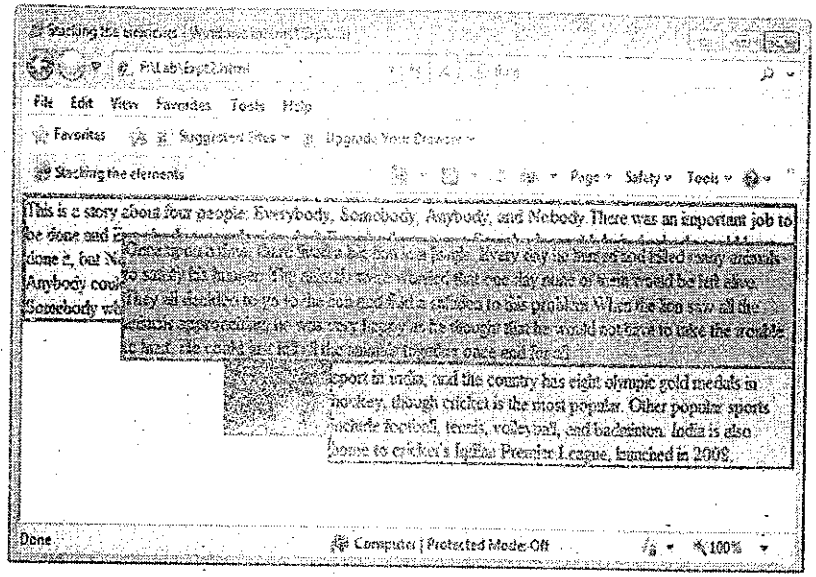
Experiment 2 : The document must contain four short paragraphs of text stacked on top of each other with only enough of each showing so that the mouse cursor can also be placed over some part of them. When the cursor is placed over the exposed part of any paragraph it should raise to the top to become completely visible.

Solution :

```
<!DOCTYPE html>
<html>
<head>
<title>Stacking the elements</title>
<script type="text/javascript">
var top="first";
function my_fun(id)
{
var Dom_obj=document.getElementById(top).style;
var Dom_new=document.getElementById(id).style;
Dom_obj.zIndex="0";
Dom_new.zIndex="30";
top=id;
}
</script>
<style type="text/css">
.text1
{
position:absolute;
top:0;
left:0;
z-index:0;
border:2px solid;
background-color:yellow;
}
.text2
{
position:absolute;
top:30px;
left:90px;
z-index:0;
border:2px solid;
background-color:aqua;
}
.text3
{
position:absolute;
top:50px;
left:180px;
```

```
z-index:0;
border:2px solid;
background-color:green;
}
.text4
{
position:absolute;
top:90px;
left:270px;
z-index:0;
border:2px solid;
background-color:pink;
}
</style>
</head>
<body>
<span class="text1" id="first" onclick="my_fun(id);">This is a story about four people: Everybody, Somebody, Anybody, and Nobody.There was an important job to be done and Everybody was asked to do it.Everybody was sure Somebody would do it.
Anybody could have done it, but Nobody did it.Somebody got angry about that because it was Everybody's job.Everybody thought Anybody could do it, but Nobody realized that Everybody wouldn't do it.It ended up that Everybody blamed Somebody when actually Nobody asked Anybody.
</span>
<span class="text2" id="second" onclick="my_fun(id);">Once upon a time, there lived a big lion in a jungle. Every day he hunted and killed many animals to satisfy his hunger. The animals were worried that one day none of them would be left alive. They all decided to go to the lion and find a solution to this problem.When the lion saw all the animals approaching, he was very happy as he thought that he would not have to take the trouble to hunt. He could just kill all the animals together once and for all.
</span>
<span class="text3" id="third" onclick="my_fun(id);">Dassera is one of the most important Hindu festivals. It is a festival that celebrates the victory of Lord Rama over Lord Ravana of Lanka. On this day, Rama had killed the great demon Ravana who had abducted Rama's wife Sita to the kingdom of Lanka. Rama, along with his brother Laxmana, Hanuman the monkey prince and his monkey army, built a bridge over the Indian ocean and crossed over to the island of Lanka. The war against Ravana lasted for ten days. Rama battled with Ravana, killed him and rescued Sita.
</span>
<span class="text4" id="fourth" onclick="my_fun(id);">
Sports in India include cricket, chess, badminton, field hockey, tennis, association football and golf. Hockey is the official national sport in India, and the country has eight olympic gold medals in hockey, though cricket is the most popular. Other popular sports include football, tennis, volleyball, and badminton. India is also home to cricket's Indian Premier League, launched in 2008.
</span>
</body>
</html>
```


Output



Experiment 3 : **Design an XML document to store information about patience in a hospital. Information about patients must include name (in 3 parts, first name, middle name, last name), social security number (SSN), age, room number, primary insurance company – including member identification number, group number and address – secondary insurance company (in the same sub parts as for the primary insurance company), known medical problems, and known drug allergies. Both attributes and nested tags must be included. Make up sample data of atleast 4 patients. Create a CSS style sheet for the above XML document and use it to create a display of that document.**

Solution :

Step 1 : Creation of XML document

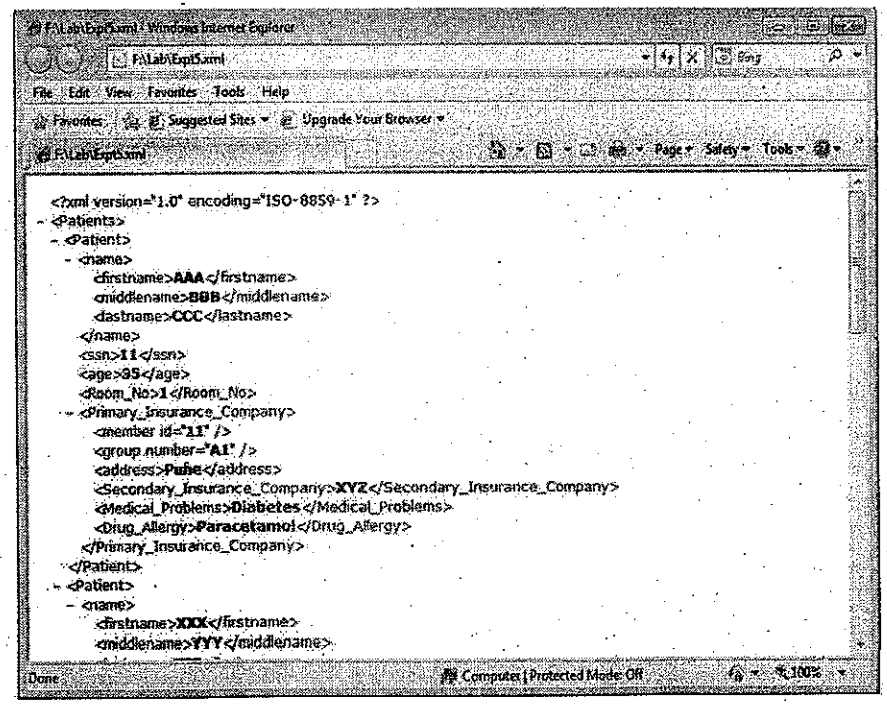
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Patients>
<Patient>
 <name>
  <firstname>AAA</firstname>
  <middlename>BBB</middlename>
  <lastname>CCC</lastname>
 </name>
 <ssn>11</ssn>
```

```
<age>35</age>
<Room_No>1</Room_No>
<Primary_Insurance_Company>
<member id="11"></member>
<group number="A1"></group>
<address>Pune</address>
<Secondary_Insurance_Company>XYZ</Secondary_Insurance_Company>
<Medical_Problems>Diabetes</Medical_Problems>
<Drug_Allergy>Paracetamol</Drug_Allergy>
</Primary_Insurance_Company>
</Patient>
<Patient>
<name>
<firstname>XXX</firstname>
<middlename>YYY</middlename>
<lastname>ZZZ</lastname>
</name>
<ssn>12</ssn>
<age>29</age>
<Room_No>2</Room_No>
<Primary_Insurance_Company>
<member id="12"></member>
<group number="B1"></group>
<address>Banglore</address>
<Secondary_Insurance_Company>KKK</Secondary_Insurance_Company>
<Medical_Problems>No</Medical_Problems>
<Drug_Allergy>No</Drug_Allergy>
</Primary_Insurance_Company>
</Patient>
<Patient>
<name>
<firstname>PPP</firstname>
<middlename>QQQ</middlename>
<lastname>RRR</lastname>
</name>
<ssn>13</ssn>
<age>45</age>
<Room_No>3</Room_No>
<Primary_Insurance_Company>
<member id="13"></member>
<group number="C1"></group>
<address>Chennai</address>
<Secondary_Insurance_Company>KMK</Secondary_Insurance_Company>
<Medical_Problems>Blood Pressure</Medical_Problems>
<Drug_Allergy>No</Drug_Allergy>
</Primary_Insurance_Company>
```



```
</Patient>
<Patient>
<name>
<firstname>FFF</firstname>
<middlename>GGG</middlename>
<lastname>HHH</lastname>
</name>
<ssn>14</ssn>
<age>15</age>
<Room_No>4</Room_No>
<Primary_Insurance_Company>
<member id="14"></member>
<group number="D1"></group>
<address>Mumbai</address>
<Secondary_Insurance_Company>KMK</Secondary_Insurance_Company>
<Medical_Problems>No</Medical_Problems>
<Drug_Allergy>No</Drug_Allergy>
</Primary_Insurance_Company>
</Patient>

</Patients>
```

Output of above step



Step 2 : Creation of CSS style sheet for the above XML document and use it to create a display of that document.

I have created a CSS file named **CSSPatient.css.** It is as follows -

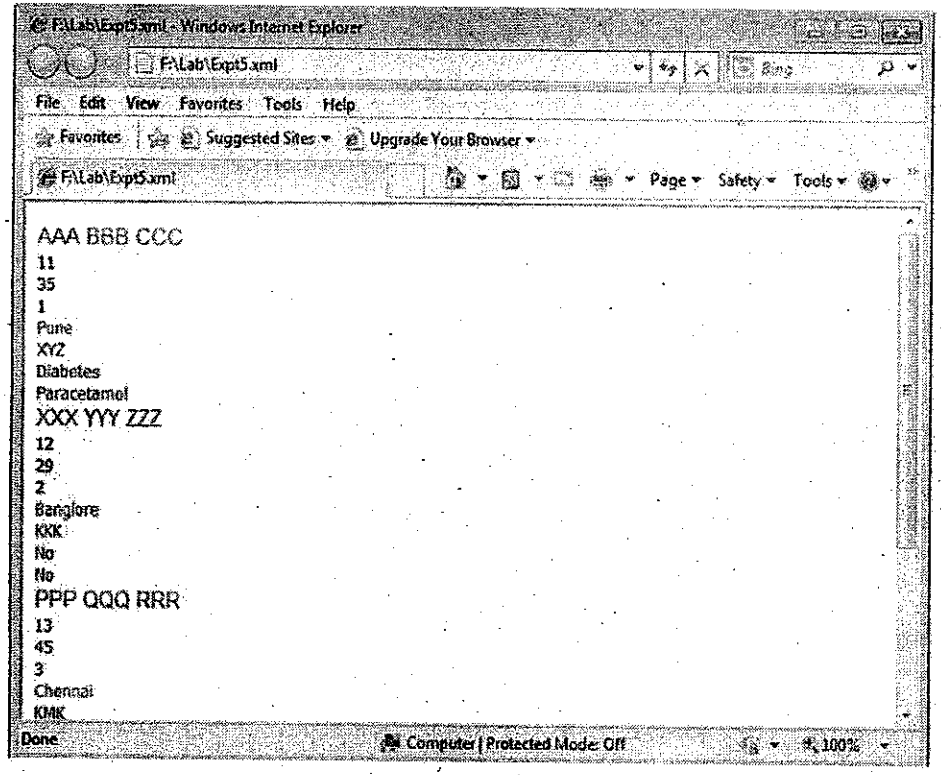
```
Patient
{
font-family:arial;
color:red;
font-size:16pt;
}
name
{
display:block;
font-family:times new roman;
font-size:14pt;
}
firstname,middlename,lastname
{
font-family:arial;
color:green;
font-size:12pt;
}
ssn,age,Room_No
{
display:block;
font-family:tahoma;
color:Blue;
font-size:10pt;
}
Primary_Insurance_Company,member,group,address
{
display:block;
font-family:tahoma;
color:green;
font-size:10pt;
}
Secondary_Insurance_Company,Medical_Problems,Drug_Allergy
{
display:block;
font-family:tahoma;
color:maroon;
font-size:10pt;
}
```

Step 3 : We will have already created a simple XML file in step 1. But we will include the path to CSS file in it. The modification will be as follows -


```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/css" href="CSSPatient.css"?>
<Patients>
 <Patient>
  <name>
   <firstname>AAA</firstname>
   <middlename>BBB</middlename>
   ...
  ...
```

Simply add this line in the XML file created in **Step 1**, rest of the program remains as it is.

Step 4 : Open the Internet Explorer and the output will be as follows -



Experiment 4 : Create the XSLT style sheet to format all the patient elements of the XML, document of exercise 3 and use it to create a display of whole element.

Solution :

Step 1 : Just modify the .xml script written in Experiment 3 , by adding a path for XSL file. The modification is as shown below -

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="XSLPatients.xsl"?>
<Patients>
<Patient>
 <name>
  <firstname>AAA</firstname>
....
....
....

</Patients>
```

Step 2 : Now write the XSL script for the above XML file. The **XSLPatinets.xsl** is as given below -

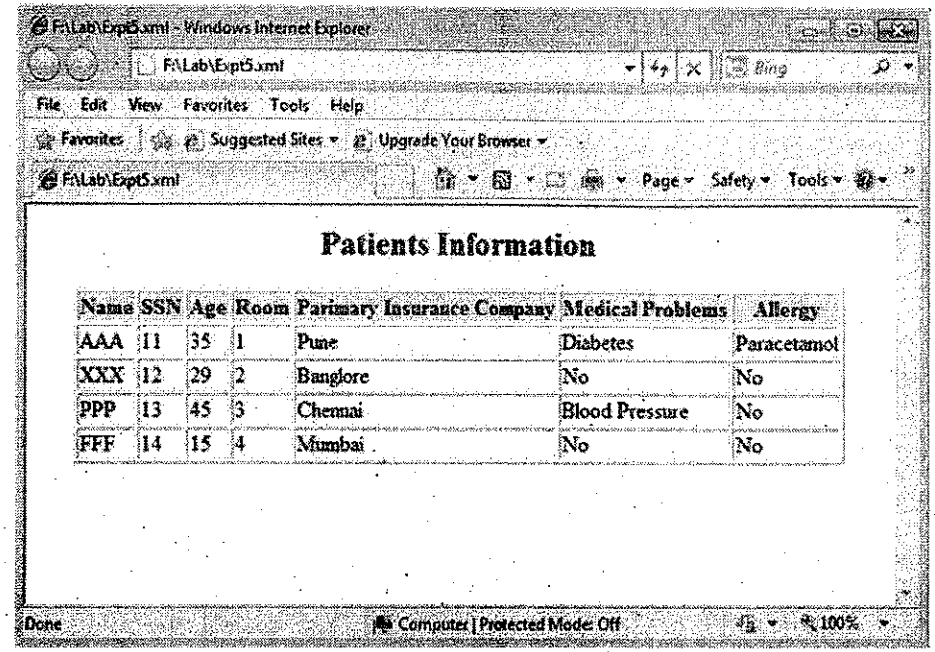
XSLPatients.xsl

```
<?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>
 <center>
 <h2>Patients Information</h2>
 <table border="1">
  <tr bgcolor="khaki">
   <th>Name</th>
   <th>SSN</th>
   <th>Age</th>
   <th>Room</th>
   <th>Parimary Insurance Company</th>
   <th>Medical Problems</th>
   <th>Allergy</th>
  </tr>
  <xsl:for-each select="Patients/Patient">
  <tr>
   <td><xsl:value-of select="name/firstname"/></td>
   <td><xsl:value-of select="ssn"/></td>
   <td><xsl:value-of select="age"/></td>
   <td><xsl:value-of select="Room_No"/></td>
   <td><xsl:value-of select="Primary_Insurance_Company/address"/></td>
   <td><xsl:value-of select="Primary_Insurance_Company/Medical_Problems"/></td>
   <td><xsl:value-of select="Primary_Insurance_Company/Drug_Allergy"/></td>
  </tr>
  </xsl:for-each>
 </table>
```



```
 </center>
 </body>
 </html>
</xsl:template>
</xsl:stylesheet>
```

Output



Experiment 5 : **Write an XHTML document to include an anchor tag, that calls a PHP document also write the called PHP document which returns a randomly chosen greeting from a list of five different greetings. The greetings must be stored as constant strings in the script. A random number between 0 and 4 can be computed with these line.**

#set the seed for mtrand with the number of microseconds

#since the last full second of the clock

mt_strand((double) microtime() * 1000000);

$number=mtrand(0,4); #computes a random integer 0-4

Write the PHP script for above to count the number of visitors and display that number for each visitor.

Hint: Use a file to store current count.

Solution :

Step 1 : Write an XHTML document, which will invoke the PHP document using an anchor tag.

The code is as follows -

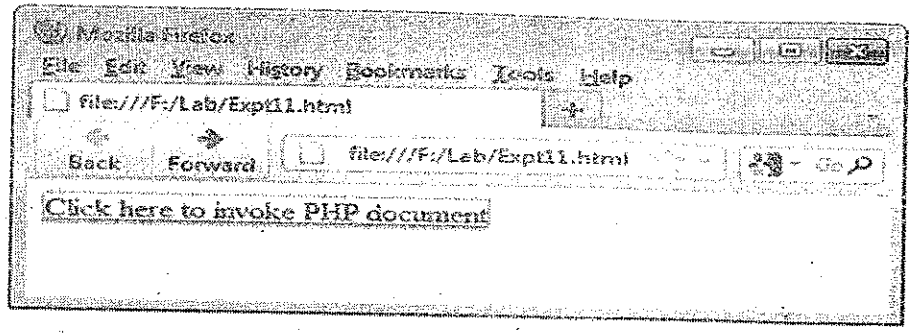
```
<html>
<head></head>
<body>
<a href="http://localhost/php-examples/test.php">Click here to invoke PHP document</a>
</body>
</html>
```

Step 2 : Now create a PHP document which generates the random number using the functions mt_srand and the function mt_rand. Note that this PHP document is invoked in step 1. Hence the name of this file is test.php. The code is as follows-

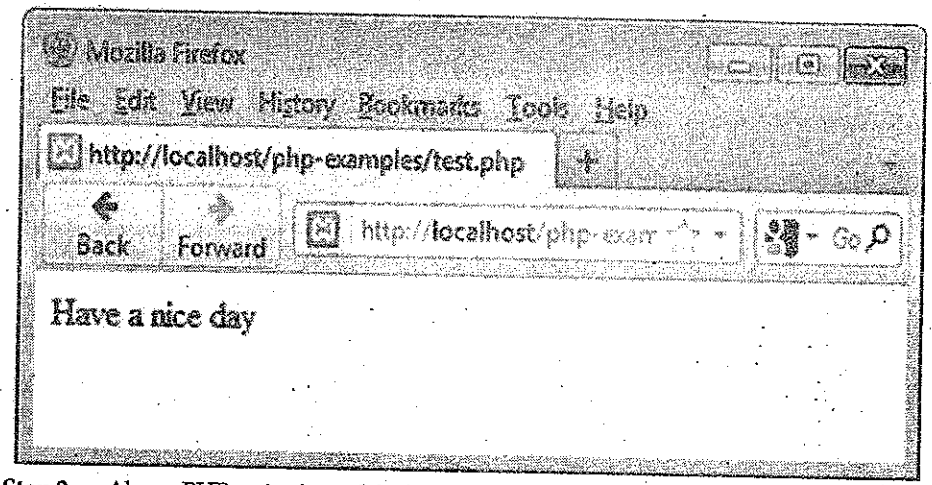
test.php

```
<?php
$str1="Good Morning";
$str2="Have a nice day";
$str3="Nice to meet you";
$str4="Get well soon";
$str5="Good bye";
function make_seed()
{
 list($usec, $sec) = explode(' ', microtime());
 return (double) $sec + ((double) $usec * 1000000);
}
mt_srand(make_seed());
$number = mt_rand(0,4);
if ($number==0)
 print $str1;
else if ($number==1)
 print $str2;
else if ($number==2)
 print $str3;
else if ($number==3)
 print $str4;
else
 print $str5;

?>
```


Output



Click on the hyperlink and you will get the PHP document invoked

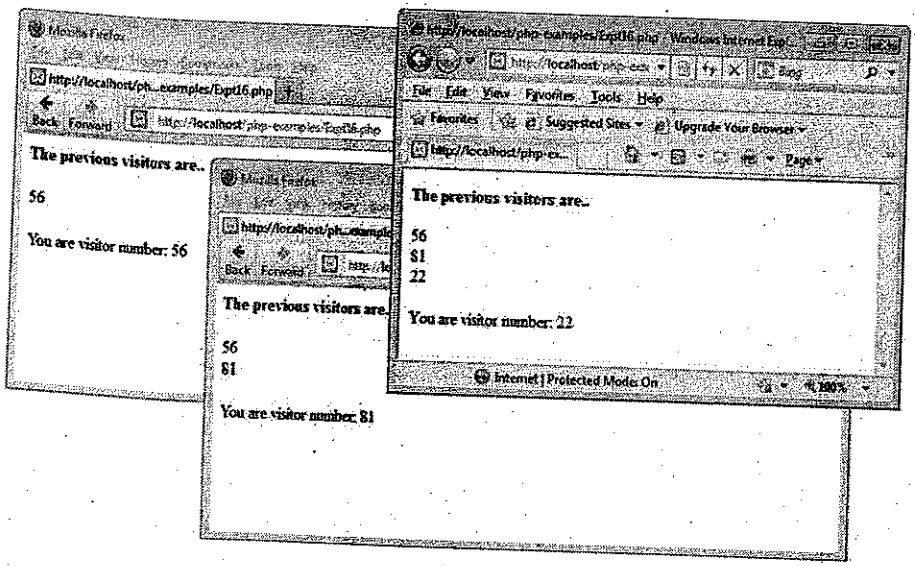


Step 3 : Above PHP script is modified to count the number of visitors and display that number for each visitor. This PHP Script is as follows –

```
<?php
function make_seed()
{
 list($usec, $sec) = explode(' ', microtime());
 return (double) $sec + ((double) $usec * 1000000);
}
mt_srand(make_seed());
$number = mt_rand(0,100);
$visit_file = 'visit.txt';
```

```
$file_handle = fopen($visit_file, 'a') or die('Cannot open file: '.$visit_file);
fwrite($file_handle, $number);
fwrite($file_handle, "\r\n");
fclose($file_handle);
$file_handle = fopen("visit.txt", "rb");
print "<h4>The previous visitors are..</h4>";
while (!feof($file_handle) )
{
$line_of_text = fgets($file_handle);
$parts = explode(',', $line_of_text);
print ("$parts[0]<br/>");
}
fclose($file_handle);
print "You are visitor number: $number";
?>
```

Output



Experiment 6 : Write the XHTML code using JavaScript Object Notation (JSON) to create the form with the following capabilities

a) A text widget to collect the users name

b) Four check boxes, one each for the following items

i) Four 100 watt light bulbs for Rs. 20=39

ii) Eight 100 watt light bulbs for Rs 40=20

iii) Four 100 watt long life light bulbs for Rs. 30=95

iv) Eight 100 watt long life light bulbs for Rs 70=49

c) A collection of 3 radio buttons that are labeled as follows

i) Visa

ii) Master Card

iii) Discover

Write a PHP script that computes the total cost of the ordered light bulbs for the above program after adding 13.5% VAT. The program must inform the buyer of exactly what was ordered in table.

Solution :

Step 1 : We will create a simple form using which the user can input the order for purchasing the items.

Expt12.html

```
<!doctype html>
<html>
<head>
<title> Order Form </title>
</head>
<body>
<form method="post" action="http://localhost/php-examples/Order.php">
<h3>Enter the Data</h3>
<table>
<tr>
<td>User Name:</td>
<td><input type="text" name="UserName"></td>
</tr>
<tr>
<td>Items:</td>
</tr>
<tr>
<td><input type="checkbox" name="option1">Four 100 Watt light bulbs for Rs 20</td>
<td><input type="text" name="val1" size="2"></td>
```

```
</tr>
<tr>
 <td><input type="checkbox" name="option2">Eight 100 Watt light bulbs for Rs.40</td>
  <td><input type="text" name="val2" size="2"></td>
</tr>
<tr>
 <td><input type="checkbox" name="option3">Four 100 Watt long life light bulbs for Rs.30</td>
  <td><input type="text" name="val3" size="2"></td>
</tr>
<tr>
 <td><input type="checkbox" name="option4">Eight 100 Watt long life light bulbs for Rs.70</td>
  <td><input type="text" name="val4" size="2"></td>
</tr>
<tr>
 <td>Payment:</td>
</tr>
<tr>
 <td><input type="radio" name="group1" value="visa">Visa</td>
</tr>
<tr>
 <td><input type="radio" name="group1" value="master">Master Card</td>
</tr>
<tr>
 <td><input type="radio" name="group1" value="discover">Discover</td>
</tr>
<tr>
 <td><input type="submit" value="Submit"></td>
</tr>
</table>
</form>

</body>
</html>
```

Step 2 : Now we will create a PHP document which reads the values submitted by the user in step 1 and will calculate the total bill.

Order.php

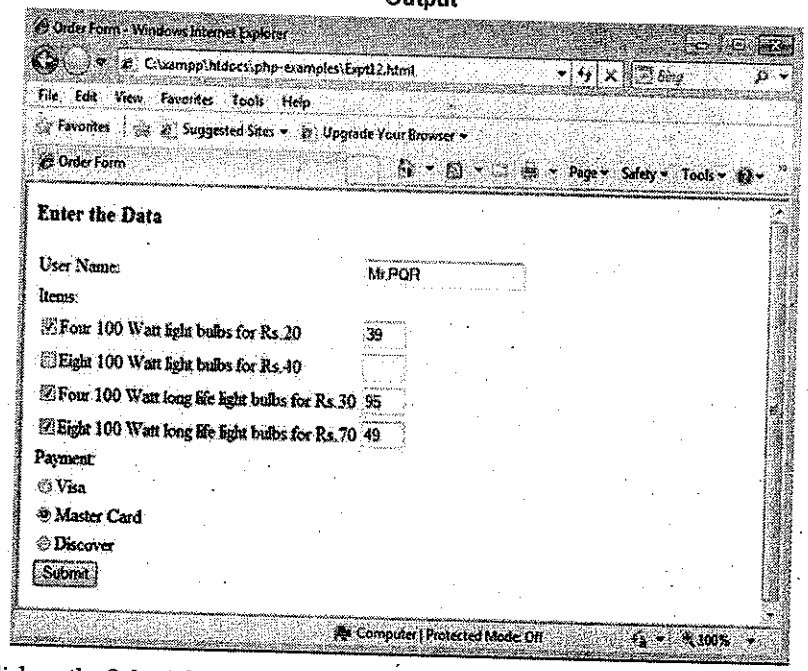
```
<html>
<head>
 <title>Display Form</title>
</head>
<body>
<?php
$UName=$_POST['UserName'];
```



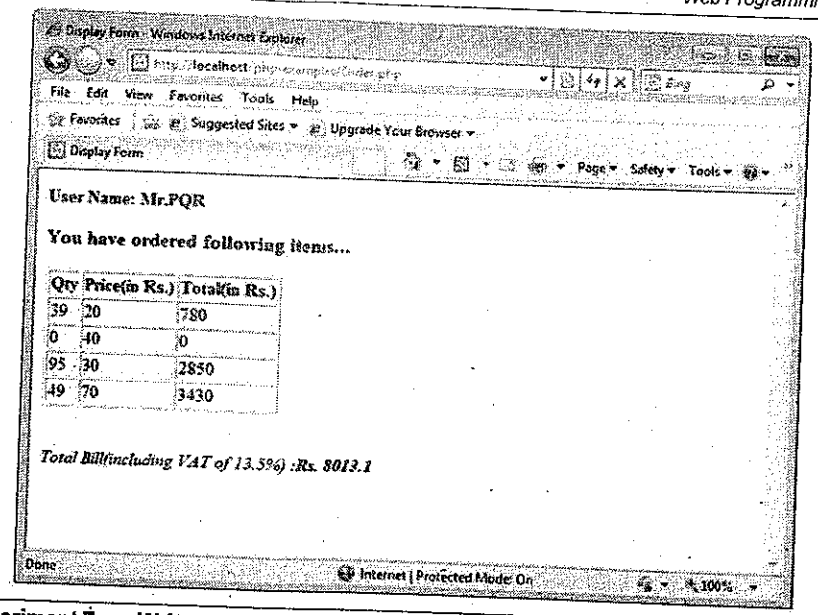
```
$Item1=0;
$Item2=0;
$Item3=0;
$Item4=0;
$price1=20;
$price2=40;
$price3=30;
$price4=70;
if (isset($_POST['option1']))
 $Item1=$_POST["val1"];
if (isset($_POST['option2']))
 $Item2=$_POST["val2"];
if (isset($_POST['option3']))
 $Item3=$_POST["val3"];
if (isset($_POST['option4']))
 $Item4=$_POST["val4"];

$total1=$Item1*$price1;
$total2=$Item2*$price2;
$total3=$Item3*$price3;
$total4=$Item4*$price4;
$total=($total1+$total2+$total3+$total4);
$total+=($total*13.5)/100;
?>
<?php print ("<h4>User Name: $UName </h4>"); ?>
<h3> You have ordered following items... </h3>
<table border=1>
<tr>
 <th>Item name</th>
 <th>Price</th>
 <th>Total</th>
</tr>
<tr>
<td> <?php print ("$Item1"); ?></td>
<td> <?php print ("$price1"); ?></td>
<td> <?php print ("$total1"); ?></td>
</tr>
<tr>
<td> <?php print ("$Item2"); ?></td>
<td> <?php print ("$price2"); ?></td>
<td> <?php print ("$total2"); ?></td>
</tr>
<tr>
<td> <?php print ("$Item3"); ?></td>
<td> <?php print ("$price3"); ?></td>
<td> <?php print ("$total3"); ?></td>
```

```
</tr>
<tr>
<td> <?php print ("$Item4"); ?></td>
<td> <?php print ("$price4"); ?></td>
<td> <?php print ("$total4"); ?></td>
</tr>
</table>
<br/><br/>
<?php print ("<i>Total Bill(including VAT of 13.5%) :<b> $total</b></i>"); ?>
</body>
</html>
```

Output



Click on the Submit button and following screen will be displayed.



Experiment 7 : Write a XHTML code to provide a form that collects names and telephone numbers. The phone numbers must be in the format ddd-ddd-dddd. Write a PHP script that checks the submitted telephone number to be sure that it confirms to the required format and then returns a response that indicates whether the number was correct.

Solution :

Step 1 : Create an HTML document which takes the name of the user as input and the phone number in required format. The HTML document is as follows -

Expt15.html

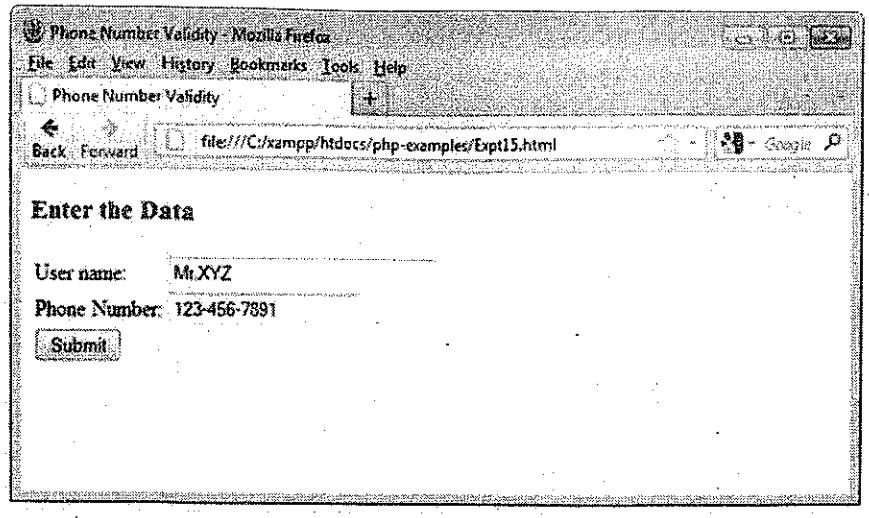
```
<html>
<head>
<title> Phone Number Validity </title>
</head>
<body>
<form method="post" action="http://localhost/php-examples/pattern.php">
<h3>Enter the Data</h3>
<table>
<tr>
<td>User name:</td>
<td><input type="text" name="UName" size="30"></td>
</tr>
<tr>
<td>Phone Number</td>
```

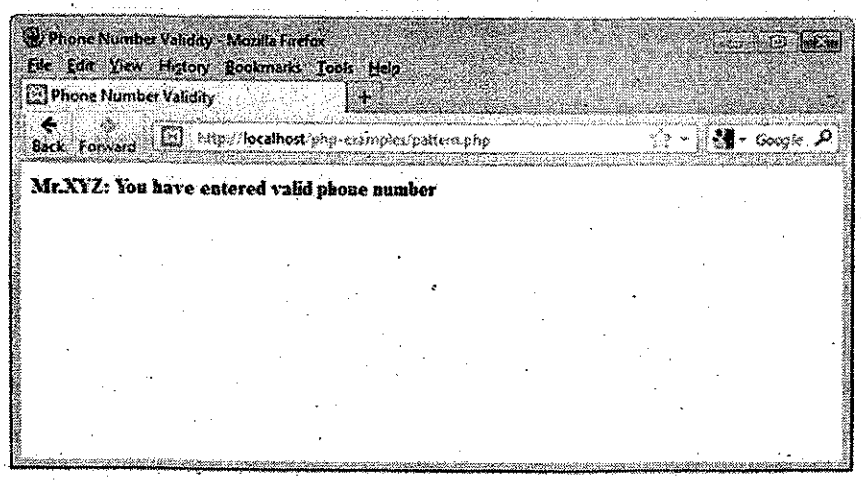
```
<td><input type="text" name="Phone"></td>
</tr>
<tr>
 <td><input type="submit" value="Submit"></td>
</tr>
</table>
</form>

</body>
</html>
```

Step 2 : Create a PHP document which checks for pattern of the phone number by comparing it with the value submitted by the user.

pattern.php

```
<html>
<head>
 <title>Phone Number Validity</title>
</head>
<body>
<?php
$UName=$_POST["UName"];
$Phone=$_POST["Phone"];
if (preg_match("/[0-9][0-9][0-9]-[0-9][0-9][0-9]-[0-9][0-9][0-9][0-9]/",$Phone))
 print "<b>$UName: You have entered valid phone number </b><br />";
else
 print "<b>The phone number is not valid</b><br />";
?>
</body>
</html>
```



Experiment 8 : **Write the XHTML code using JavaScript Object Notation (JSON) to accept from the user name, phone no, mail-id, stored in database. Retrieve same information from database using a separate PHP script.**

Solution :

Step 1 : Create an XHTML document as follows -

input.html

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

```
        <title> Inserting Data into Database Table</title>
</head>
<body>
        <form method = "post" action = "http://localhost/php-examples/insertData.php ">
        <label>Name</label>
        <input type="text" name="txt_name" />
        <br />
        <label>City</label>
        <input type="text" name="txt_city" />
        <br />
        <label>Phone Number</label>
        <input type="text" name="txt_phone" />
        <br />
        <label>Email</label>
        <input type="text" name="txt_email" />
        <br />
        <input type="submit" value="Submit">
</form>
</body>
</html>
```

Step 2 : Now create PHP script named **insertData.php** for accepting the values submitted by above HTML code. This php script inserts these accepted values into the database created using MYSQL.

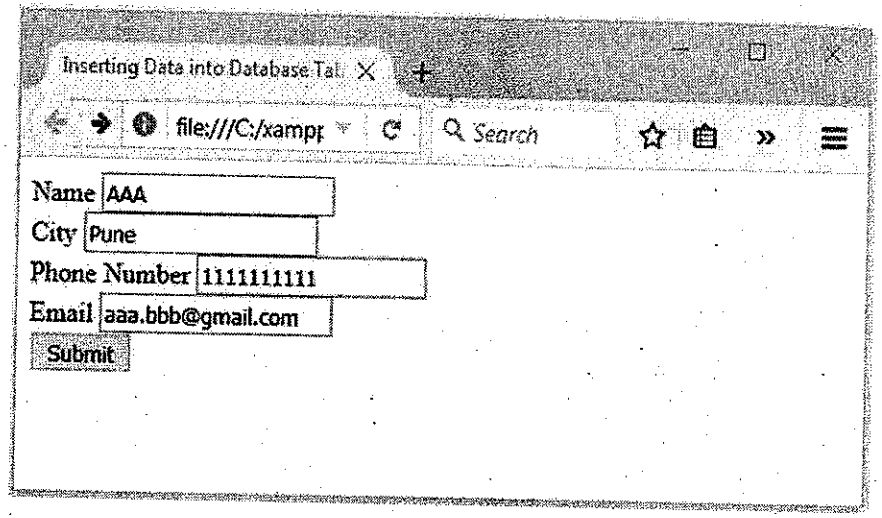
insertData.php

```
<?php
// Make a MySQL Connection
$con=mysql_connect("localhost","root","");
$name=$_POST['txt_name'];
$city=$_POST['txt_city'];
$ph=$_POST['txt_phone'];
$email=$_POST['txt_email'];
if(!$con)
{
die('error in connection'.mysql_error());
}
mysql_select_db('Person',$con); //select the database
$query=" INSERT INTO my_detail(name,city,phone_no,mail_id)
VALUES('$name','$city','$ph','$email')";
mysql_query($query,$con);//Execution of Query
//Execution of Query for displaying the data
echo "<h3> Displaying Records Line by Line </h3>";
$result=mysql_query("SELECT * FROM my_detail");
while($row = mysql_fetch_array($result))
  {
  echo $row['name'] . " " . $row['city'] . " " . $row['phone_no'] . " " . $row['mail_id'] ;
```

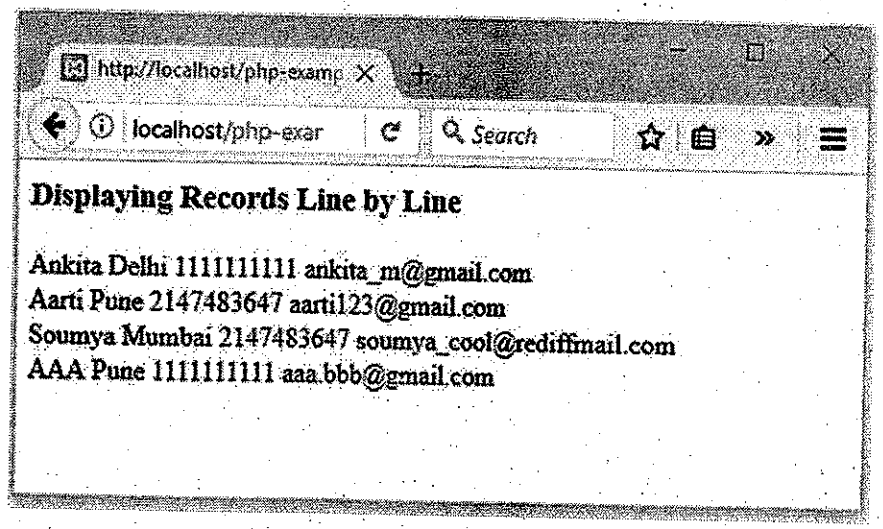


```
                        //Each record will be displayed
                        //line by line
 echo "<br />";
 }
 mysql_close($con);     //closing the database
?>
```

Step 3 : Start apache, Open the web browser and execute the HTML document created in step 1.



Click submit button.

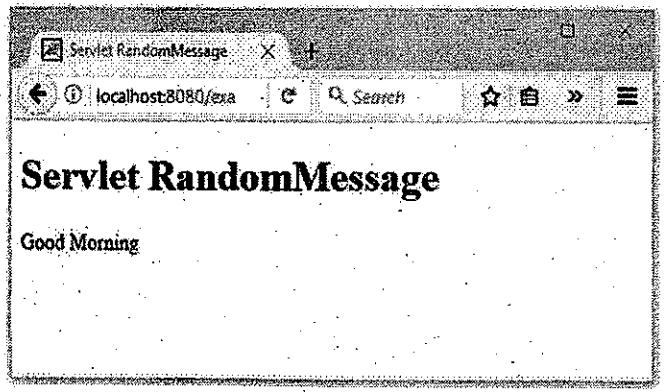


Experiment 9 : Write a servlet that returns a randomly chosen greeting from a list of five different greetings. The greeting must be stored as constant strings in the program.

Solution :

```
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Greetings extends HttpServlet
{
        public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException
        {
                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                String[] myStrings = {"Good Morning", "Good night","How are You?", "Good
Evening", "See you"};
                out.println("<!DOCTYPE html>");
                out.println("<html>");
                out.println("<head>");
                out.println("<title>Servlet RandomMessage</title>");
                out.println("</head>");
                out.println("<body>");
                out.println("<h1>Servlet RandomMessage</h1>");
                Random r = new Random();
                out.println(myStrings[r.nextInt(myStrings.length)]);
                out.println("</body>");
                out.println("</html>");
        }
}
```

Output



Experiment 10 : Write a servlet for the XHTML code of exercise 6 that computes the total cost of ordered light bulbs after adding 2% sales tax. The servlet must inform the buyer of exactly what was ordered in table.

Solution :

Step 1 : The XHTML code for inputting the values is

```
<!doctype html>
<html>
<head>
<title> Order Form </title>
</head>
<body>
<form action="http://localhost/examples/servlets/servlet/Order">
<h3>Enter the Data</h3>
<table>
<tr>
<td>User Name:</td>
<td><input type="text" name="UserName"></td>
</tr>
<tr>
<td>Items:</td>
</tr>
<tr>
<td><input type="checkbox" name="option1">Four 100 Watt light bulbs for Rs.20</td>
<td><input type="text" name="val1" size="2"></td>
</tr>
<tr>
<td><input type="checkbox" name="option2">Eight 100 Watt light bulbs for Rs.40</td>
<td><input type="text" name="val2" size="2"></td>
</tr>
<tr>
<td><input type="checkbox" name="option3">Four 100 Watt long life light bulbs for
Rs.30</td>
<td><input type="text" name="val3" size="2"></td>
</tr>
<tr>
<td><input type="checkbox" name="option4">Eight 100 Watt long life light bulbs for
Rs.70</td>
<td><input type="text" name="val4" size="2"></td>
</tr>
<tr>
<td>Payment:</td>
</tr>
<tr>
<td><input type="radio" name="group1" value="visa">Visa</td>
```

```
</tr>
<tr>
 <td><input type="radio" name="group1" value="master">Master Card</td>
</tr>
<tr>
 <td><input type="radio" name="group1" value="discover">Discover</td>
</tr>
<tr>
 <td><input type="submit" value="Submit"></td>
</tr>
</table>
</form>
</body>
</html>
```

Step 2 : The servlet code for computing the bill is as follows

Order.java

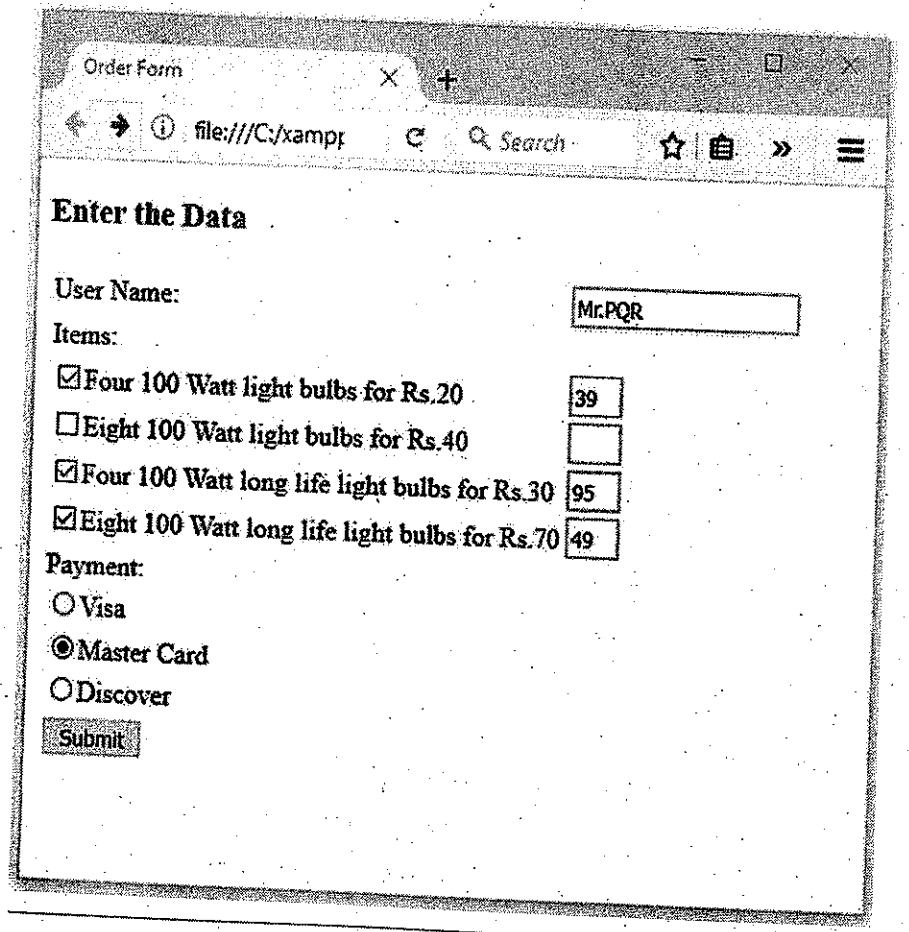
```
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Order extends HttpServlet
{
        public void doGet(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException
        {
                res.setContentType("text/html");
                PrintWriter out = res.getWriter();
                String UName=req.getParameter("UserName");
                int Item1,Item2,Item3,Item4;
                int price1=20;
                int price2=40;
                int price3=30;
                int price4=70;
                Item1=Item2=Item3=Item4=0;
                if(req.getParameter("option1")!=null)
                {
                        String str1=req.getParameter("val1");
                        Item1 =Integer.parseInt(str1);
                }
                if(req.getParameter("option2")!=null)
                {
                        String str2=req.getParameter("val2");
                        Item2 =Integer.parseInt(str2);
                }
                if(req.getParameter("option3")!=null)
```

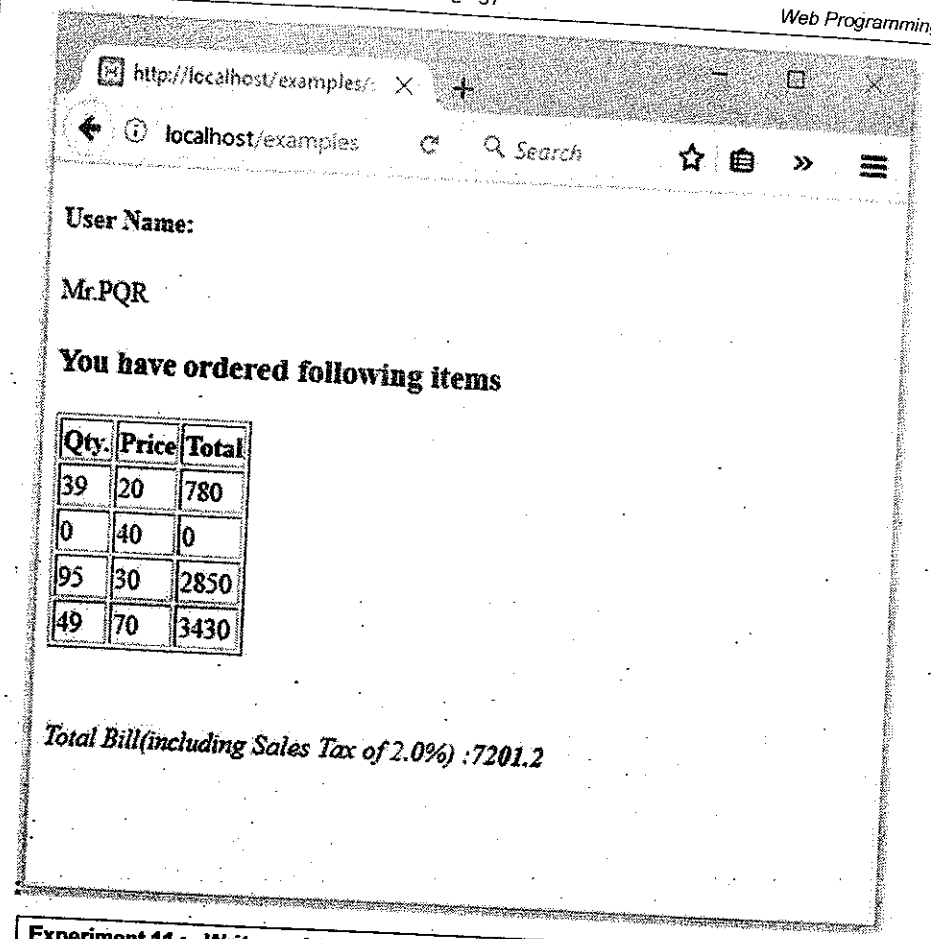


```
                {
                        String str3=req.getParameter("val3");
                        Item3 =Integer.parseInt(str3);
                }
                if(req.getParameter("option4")!=null)
                {
                        String str4=req.getParameter("val4");
                        Item4 =Integer.parseInt(str4);
                }
                int total1=Item1*price1;
                int total2=Item2*price2;
                int total3=Item3*price3;
                int total4=Item4*price4;
                double total=(total1+total2+total3+total4);
                total+=(total*2.0)/100;
                out.println("<!DOCTYPE html>");
                out.println("<html>");
                out.println("<head>");
                out.println("</head>");
                out.println("<body>");
                out.println ("<h4>User Name:</h4>"+UName);
                out.println ("<h3>You have ordered following items </h3>");
                out.println("<table border=1>");
                out.println("<tr>");
                out.println("<th>Qty.</th>");
                out.println("<th>Price</th>");
                out.println("<th>Total</th>");
                out.println("</tr>");

                out.println("<tr>");
                out.println("<td>"+Item1+"</td>");
                out.println("<td>"+price1+"</td>");
                out.println("<td>"+total1+"</td>");
                out.println("</tr>");

                out.println("<tr>");
                out.println("<td>"+Item2+"</td>");
                out.println("<td>"+price2+"</td>");
                out.println("<td>"+total2+"</td>");
                out.println("</tr>");

                out.println("<tr>");
                out.println("<td>"+Item3+"</td>");
                out.println("<td>"+price3+"</td>");
                out.println("<td>"+total3+"</td>");
                out.println("</tr>");
```

```
            out.println("<tr>");
            out.println("<td>"+item4+"</td>");
            out.println("<td>"+price4+"</td>");
            out.println("<td>"+total4+"</td>");
            out.println("</tr>");
            out.println("</table><br/><br/>");
            out.println("<i>Total Bill(including Sales Tax of 2.0%) :
                        <b>"+total+"</b></i>");
            out.println("</body>");
            out.println("</html>");
        }
}
```

Output





Experiment 11 : Write and test a JSP document that displays the form of exercise 6 and produces the same response document as exercise 10.

Solution :

Step 1 : Create an XHTML form for inputting the values

```
<!doctype html>
<html>
<head>
<title> Order Form </title>
</head>
<body>
<form action="http://localhost/examples/jsp/BillDemo/Order.jsp">
<h3>Enter the Data</h3>
<table>
```

```
<tr>
<td>User Name:</td>
<td><input type="text" name="UserName"></td>
</tr>
<tr>
 <td>Items:</td>
</tr>
<tr>
 <td><input type="checkbox" name="option1">Four 100 Watt light bulbs for Rs.20</td>
 <td><input type="text" name="val1" size="2"></td>
</tr>
<tr>
 <td><input type="checkbox" name="option2">Eight 100 Watt light bulbs for Rs.40</td>
  <td><input type="text" name="val2" size="2"></td>
</tr>
<tr>
 <td><input type="checkbox" name="option3">Four 100 Watt long life light bulbs for
Rs.30</td>
  <td><input type="text" name="val3" size="2"></td>
</tr>
<tr>
 <td><input type="checkbox" name="option4">Eight 100 Watt long life light bulbs for
Rs.70</td>
  <td><input type="text" name="val4" size="2"></td>
</tr>
<tr>
 <td>Payment:</td>
</tr>
<tr>
 <td><input type="radio" name="group1" value="visa">Visa</td>
</tr>
<tr>
 <td><input type="radio" name="group1" value="master">Master Card</td>
</tr>
<tr>
 <td><input type="radio" name="group1" value="discover">Discover</td>
</tr>
<tr>
 <td><input type="submit" value="Submit"></td>
</tr>
</table>
</form>
</body>
</html>
```

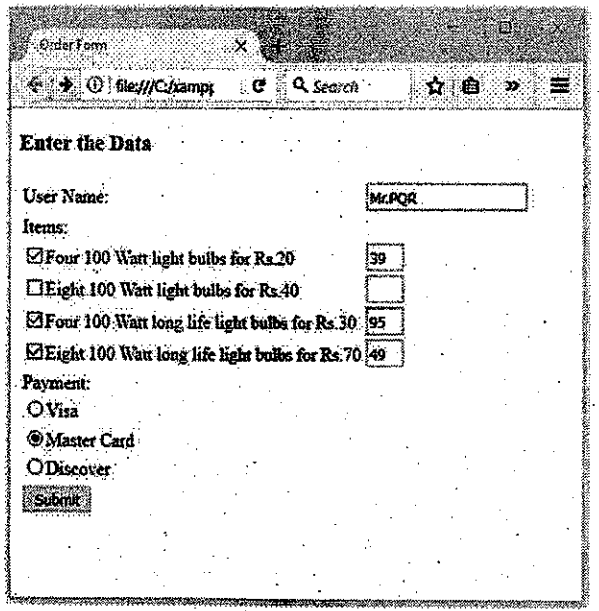
Step 2 : Create an JSP document and compute the total bill by accepting the values from above HTML document.

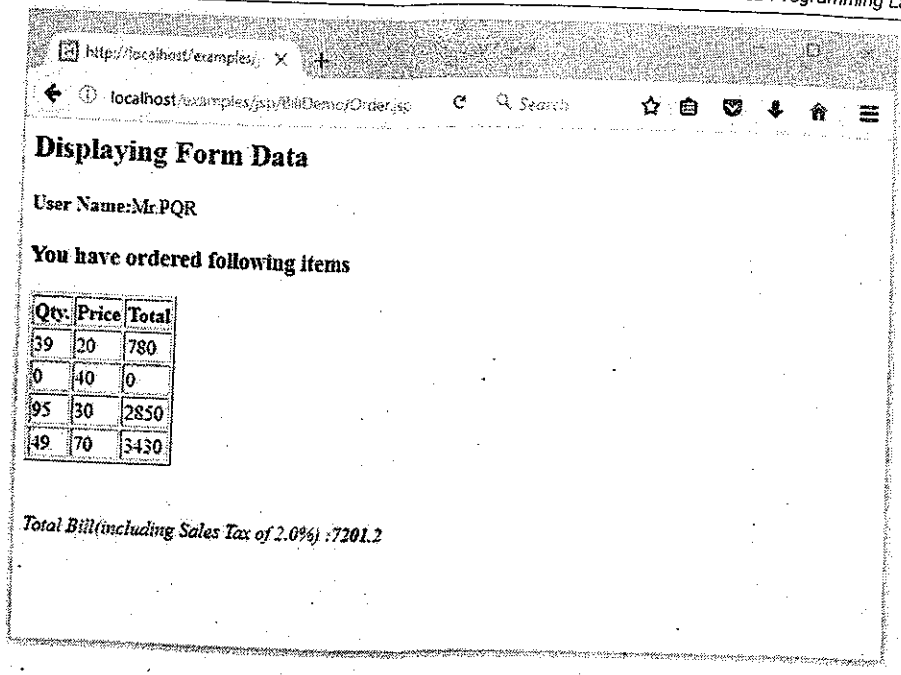
Order.jsp

```
<html>
<body>
<h2>Displaying Form Data</h2>
<b>User Name:</b><%= request.getParameter("UserName")%><br/>
<%
        int Item1,Item2,Item3,Item4;
        int price1=20;
        int price2=40;
        int price3=30;
        int price4=70;
        Item1=Item2=Item3=Item4=0;
        if(request.getParameter("option1")!=null)
        {
                String str1=request.getParameter("val1");
                Item1 =Integer.parseInt(str1);
        }
        if(request.getParameter("option2")!=null)
        {
                String str2=request.getParameter("val2");
                Item2 =Integer.parseInt(str2);
        }
        if(request.getParameter("option3")!=null)
        {
                String str3=request.getParameter("val3");
                Item3 =Integer.parseInt(str3);
        }
        if(request.getParameter("option4")!=null)
        {
                String str4=request.getParameter("val4");
                Item4 =Integer.parseInt(str4);
        }
        int total1=Item1*price1;
        int total2=Item2*price2;
        int total3=Item3*price3;
        int total4=Item4*price4;
        double total=(total1+total2+total3+total4);
        total+=(total*2.0)/100;
%>
        <h3>You have ordered following items </h3>
        <table border=1>
        <tr>
                <th>Qty.</th>
                <th>Price</th>
                <th>Total</th>
        </tr>
```

```
            <tr><td><%=Item1%></td>
            <td><%=price1%></td>
            <td><%=total1%></td>
        </tr>
        <tr>
            <td><%=Item2%></td>
            <td><%=price2%></td>
            <td><%=total2%></td>
        </tr>
        <tr>
            <td><%=Item3%></td>
            <td><%=price3%></td>
            <td><%=total3%></td>
        </tr>
        <tr>
            <td><%=Item4%></td>
            <td><%=price4%></td>
            <td><%=total4%></td>
        </tr>

    </table><br/><br/>
    <i>Total Bill(including Sales Tax of 2.0%) :<b><%=total%></b></i>
</body>
</html>
```

Output





Experiment 12 : Write a markup document to create a form that collects favourite popular songs, including the name of the song, the composer and the performing artist or group. This document must call a servlet when the form is submitted and another servlet to request a current list of survey results.

Solution :

Step 1 : The markup document that will collect the favourite popular songs is as follows -

inputSong.html

```
<!doctype html>
<html>
<head>
<title>Songs Survey </title>
</head>
<body>
<form action="http://localhost/examples/servlets/servlet/SongsList">
<h3>Enter the Data</h3>
<table>
<tr>
<td>Name of the Song:</td>
<td><input type="text" name="SongName" size="30"></td>
```

Notes

Solved Model Question Bank
Web Programming
Semester - V (Computer Science and Engineering)

Unit - I

Q.1 *Define Web Browser, Web Server, Event, Event Handling and Internet.* **(Refer sections 1.3, 1.4, 2.5 and 1.1.2)** **[5]**

Q.2 *Explain domain names with an example.* **(Refer section 1.1.4)** **[5]**

Q.3 *Explain the operation of web server.* **(Refer section 1.4.1)** **[5]**

Q.4 *Illustrate the general server characteristics.* **(Refer section 1.4.2)**

Q.5 *Explain the file structure of web server.* **(Refer section 1.4.2)** **[5]**

Q.6 *Define MIME, Web security, URL* **(Refer sections 1.6, 1.5 and 1.8)** **[5]**

Q.7 *Discuss URL format with its different paths.* **(Refer section 1.5)** **[5]**

Q.8 *Explain MIME with its type specification.* **(Refer section 1.6)** **[5]**

Q.9 *Illustrate the HTTP protocol's request and response phases.* **(Refer section 1.7)** **[5]**

Q.10 *Discuss web security issues.* **(Refer section 1.8)** **[5]**

Q.11 *Mention the differences between html and xhtml.* **(Refer section 1.10.1)** **[5]**

Q.12 *Explain the HTTP protocol's request and response phases with an example for each.* **(Refer section 1.7)** **[10]**

Q.13 *Explain the operation of Web Server with its file structure.* **(Refer section 1.4.2)** **[10]**

Unit - II

Q.14 *Explain the structure of DOM.* **(Refer section 2.3)** **[5]**

Q.15 *List DOM node properties.* **(Refer section 2.3)** **[5]**

Q.16 *Describe the parameters and actions of the setTimeout and setInterval functions.* **(Refer section 3.10)** **[5]**

Q.17 *Illustrate Moving elements with simple example.* **(Refer section 3.10)** **[5]**

Q.18 *Explain Element visibility with simple example.* **(Refer section 3.4)** **[5]**

Q.19 *Explain how to locate the mouse cursor.* **(Refer section 3.8)** **[5]**

Q.20 *Explain different types of positioning with an example.* **(Refer section 3.2)** **[10]**

Q.21 *Illustrate with an example for dynamic stacking of images.* **(Refer section 3.7)** [10]

Q.22 *Explain with an example how to change dynamically background and foreground Colors of the document.* **(Refer section 3.5.1)** [10]

Q.23 *Illustrate moving elements and element visibility with an example.* **(Refer sections 3.4 and 3.10)** [10]

Q.24 *Discuss the two ways to register an event handler in DOM-0 event model.* **(Refer section 2.9.2)** [10]

Q.25 *Explain the 3 phases of event processing in the DOM-2 event model.* **(Refer section 2.9.1)** [10]

Q.26 *Illustrate how to handle blur event and change event with an example.* **(Refer section 3.6)** [10]

Q.27 *Explain how to handle the focus event with an example.* **(Refer section 2.8)** [10]

Unit - III

Q.28 *Write a note on XML.* **(Refer section 4.2)** [5]

Q.29 *List the features of XML* **(Refer section 4.2.1)** [5]

Q.30 *Write a note on XML document structure.* **(Refer section 4.3)** [5]

Q.31 *Define DTD. Mention four possible keywords in DTD declaration.* **(Refer section 4.4)** [5]

Q.32 *Define XML schema. List the advantages of XML schema over DTD.* **(Refer section 4.6.1)** [5]

Q.33 *Differentiate between simple type and complex type XML elements.* **(Refer sections 4.6.5 and 4.6.6)** [5]

Q.34 *Explain Internal and External DTD's with an example.* **(Refer section 4.4)** [5]

Q.35 *Illustrate declaring of elements, attributes and entities in DTD with an example.* **(Refer section 4.4)** [10]

Q.36 *Explain different XSD indicators.* **(Refer section 4.6.6)** [10]

Q.37 *Explain how to declare namespace with example.* **(Refer section 4.5)** [10]

Q.38 *Explain schema and schema instance with an example.* **(Refer section 4.6)** [10]

Q.39 *Explain how to create simple type and complex type element with an example.* **(Refer sections 4.6.5 and 4.6.6)** [10]

Unit - IV and V

Q.40 *Write a note on PHP.* **(Refer section 5.2)** [5]

Q.41 *Explain four scalar types of PHP.* **(Refer section 5.4)** [5]

Q.42 *Explain built-in string manipulation functions.* **(Refer section 5.4.7)** [5]

Q.43 *Explain Implicit and Explicit type conversions.* **(Refer section 5.4.8)** [5]

Q.44 *Write PHP script to compute the sum of positive integers upto 30 using do-while statement.* **(Refer example 5.6.3)** [5]

Q.45 *Write PHP script to compute factorial of 'n' using while or for loop construct.* **(Refer example 5.6.4)** [5]

Q.46 *Explain the syntax of for-each statement with an example.* **(Refer section 5.6)** [5]

Q.47 *Write a PHP script to sort an array of elements.* **(Refer section 5.7.5)** [5]

Q.48 *Explain how cookies are extracted using PHP.* **(Refer section 5.12)** [5]

Q.49 *Write a note on session tracking.* **(Refer section 5.13)** [5]

Q.50 *Explain preg_match() and preg_split() functions with example.* **(Refer section 5.9)** [5]

Q.51 *Explain potential problems associated with special characters.* **(Refer section 6.1.1)** [5]

Q.52 *Write PHP functions used to connect to MYSQL database and selecting a database* **(Refer section 6.1.2)** [5]

Q.53 *Construct a PHP script to insert records through form into a database.* **(Refer section 6.1.3(6))** [5]

Q.54 *Construct a PHP script to retrieve records from the database table.* **(Refer section 6.1.3(7))** [5]

Q.55 *Discuss how the collection of metadata extracted from a database.* **(Refer section 6.2.7)** [5]

Q.56 *Explain the use of mysql_query() method used to execute SQL queries with its syntax.* **(Refer section 6.2)** [5]

Q.57 *Construct a PHP script to compute the squareRoot, Square, Cube and Quad of 10 numbers.* **(Refer example 5.6.6)** [10]

Q.58 *Explain how to create indexed and associated array with an example.* **(Refer section 5.7.1)** [10]

Q.59 *Write a PHP script to illustrate sort, assort and ksort functions.* **(Refer section 5.7.5)** [10]

Q.60 *Write a note on PHP and its scalar types.* **(Refer section 5.4)** [10]

Q.61 *Explain types of arrays with an example in PHP.* **(Refer section 5.7.1)** [10]

Q.62 *Write a PHP script to create a table, insert records into the table, retrieve records from the table. Assume database "Student" and table "cs_student" with fields name, sem, regno, address,* **(Refer example 6.1.1)** [10]

Q.63 *Write a PHP script to insert record into the table and retrieve records from the table. Assume a table "my_detail" is already created with fields name, city, phone_no and mail_id.* **(Refer example 6.1.2)** [10]

Q.64 *Construct a PHP script to insert and retrieve records from the database table.*

Q.65 *Illustrate the use of :*

Mysql_query(), mysql_connect(), mysql_select_db(), mysql_num_rows() and mysql_num_fields() **(Refer section 6.1)** [10]

Q.66 *Explain the steps involved in accessing mySQL database through JDBC.* **(Refer section 6.2.6)** [10]

Unit - VI

Q.66 *Write a note on Servlet Containers.* **(Refer section 7.3)** [5]

Q.67 *Explain doGet and doPost methods of the HttpServlet class.* **(Refer section 7.2)** [5]

Q.68 *Write a note on Cookies.* **(Refer section 7.5.1)** [5]

Q.69 *List the five parts of JSTL.* **(Refer section 7.9)** [5]

Q.70 *Explain three elements associated with JSP.* **(Refer section 7.6.2)** [5]

Q.71 *Write a note on MVC application Architecture.* **(Refer section 7.11)** [5]

Q.72 *Explain the two standard tag libraries of JSF.* **(Refer section 7.12)** [5]

Q.73 *Explain the different methods of Cookies with an example.* **(Refer section 7.5.1)** [10]

Q.74 *Explain the processing flow of JSP documents with a neat diagram.* **(Refer section 7.6.3)** [10]

Q.75 *Illustrate JSTL control action elements with an example for each.* **(Refer section 7.9)** [10]

❑❑❑

Solved Model Question Paper
Web Programming

Semester - V (Computer Science and Engineering)

Time : 3 Hours **Max Marks : 100**

PART - A

Answer any SIX questions. Each carries 5 marks. **[5 × 6 = 30 Marks]**

Q.1 *Define Web Browser, Web Server, Event, Event Handling and Internet.* **(Refer sections 1.3, 1.4, 2.5 and 1.1.2)**

Q.2 *Describe the parameters and actions of the setTimeout and setInterval functions.* **(Refer section 3.10)**

Q.3 *Define DTD. Mention four possible keywords in DTD declaration.* **(Refer section 4.4)**

Q.4 *Explain four scalar types of PHP.* **(Refer section 5.4)**

Q.5 *Explain built-in string manipulation functions.* **(Refer section 5.4.7)**

Q.6 *Explain potential problems associated with special characters.* **(Refer section 6.1.1)**

Q.7 *Construct a PHP script to insert records through form into a database.* **(Refer section 6.1.3(6))**

Q.8 *Write a note on JavaBeams.* **(Refer section 7.10)**

Q.9 *Explain JSF event handling.* **(Refer section 7.14)**

PART - B

Answer any SEVEN full questions. Each carries 10 marks. **[10 × 7 = 70 Marks]**

Q.1 *Illustrate the HTTP protocol's request and response phases with an example for each.* **(Refer section 1.7)**

Q.2 *Illustrate with an example for dynamic stacking of images.* **(Refer section 3.7)**

Q.3 *Explain declaring of elements, attributes and entities in DTD with an example.* **(Refer section 4.4)**

Q.4 *Explain different XSD indicators.* **(Refer section 4.6.6)**

Q.5 *Explain how to create indexed and associated array with an example.* **(Refer section 5.7.1)**

Q.6 *Write a PHP script to illustrate sort, assort and ksort functions.* **(Refer section 5.7.5)**

Q.7 *Write a PHP script to insert record into the table and retrieve records from the table. Assume a table "my_detail" is already created with fields name, city, phone_no and mail_id.* **(Refer section 6.1.2)**

Q.8 *Explain the steps involved in accessing mySQL database through JDBC.* **(Refer section 6.2.6)**

Q.9 *Explain life cycle of a Servlet.* **(Refer section 7.1.2)**

Q.10 *Explain steps in JDBC.* **(Refer example 6.2)**

❑❑❑