1.1 Basics of WWW

- **World Wide Web**(WWW) is a collection 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.

1.1.1 History of WWW

- The concept of WWW was introduced by **Sir Tim Berners-Lee** at the **European Organization for Nuclear Research (CERN).** 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.
- During 1992-1995, along with **HTTP protocol** a new protocol named Gopher **protocol** came up 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. 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. 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 HTTP Protocol Methods and Headers

GTU : Summer-11,12,13,15

Hyper TextTransfer Protocol (HTTP) takes part in web browser and web server communication. Hence it is called a communication protocol. The basic feature of HTTP

Scanned with CamScanner

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.

1.2.1 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.

Scanned with CamScanner

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 **U**niform **R**esource **I**dentifier(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 **U**niform **R**esource **L**ocator and URN stands for **U**niform **R**esource **N**ame. 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

Scanned with CamScanner

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 -

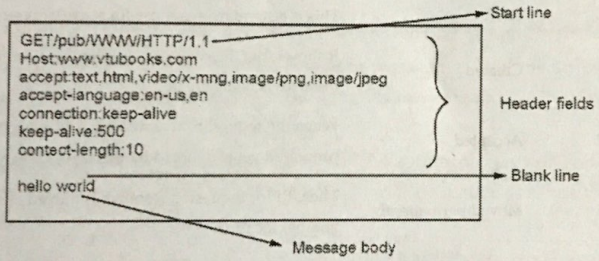


Fig. 1.2.1 HTTP request message structure

1.2.2 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.

Scanned with CamScanner

- There are various scripting languages available for displaying the web page on the web browser. These scripting languages are categorised into client side scripting and server side scripting. The popular client side scripting languages –HTML,XML, JavaScript,DHTML, PHP and the server side scripting languages are JSP,ASP, Servlets and so on. These web documents are useful for displaying the text and image on the web browsers.

1.3.1 Functions Defined by Web Browser

Various functions of web browser are -

1. Reformat the URL and send a valid HTTP request.
2. When user gives the address of particular web site it is in the form of domain name. The web browser coverts the DNS to corresponding IP address.
3. The web browser establishes a connection with the Web browser while processing the user's request.
4. The web browsers send the HTTP request to the web server.
5. The web server processes the HTTP request sent by the web browser and returns the desired web page to the client machine. The web browser on the client's machine displays this web page in appropriate format.
6. Most of the web browsers automatically store(Cache) the recently visited web pages. This feature is called **Cache control**.

1.4 Architecture of Web Browser

GTU : Summer -12

The web browser architecture consists of various layers such as Parser/Interpreter Layer, Rendering Engine, Browser Engine, and user interface. The user interface is the top most layer which contains the web browser. The web browser has various menus/tools such as Address bar, Bookmarks, Settings and so on. The user executes web page in the web browser. For executing this web document sometimes the plugins or Add-ons are required. These are supported by the browser engine. The data cache stores the recently referred web documents. At the base of this architecture is the Interpreter/Parser layer. The Rendering Engine submits the web documents to this layer. If the web document is XML document then XML parser parses it. If any error in the script is present then it is highlighted. In this layer the web document is processed for checking its validity.

Scanned with CamScanner

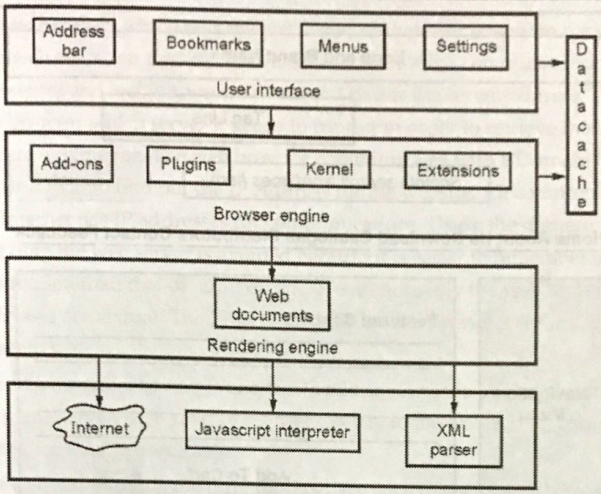


Fig. 1.4.1 Browser architecture

Web Site Structure

- The Web site structure can be created by considering major web pages of the website.
- In the research phase the **design team** starts working on the look and feel of the web pages. The **technical team** gathers and analyses the technology infrastructure of organization. It also find out the limitations and constraints of the intended audience. The design sketches pool the collective knowledge of design team, technical team and information architect.
- It is a simple design sketch created for the website of E-marketing. The information of particular product can be viewed on the website. For purchasing it, user can add it to cart.

See Fig. 1.4.2 on next page.

Review Question

1. *Explain the architecture of the browser.* **GTU : Summer -12, Marks 3**

Scanned with CamScanner

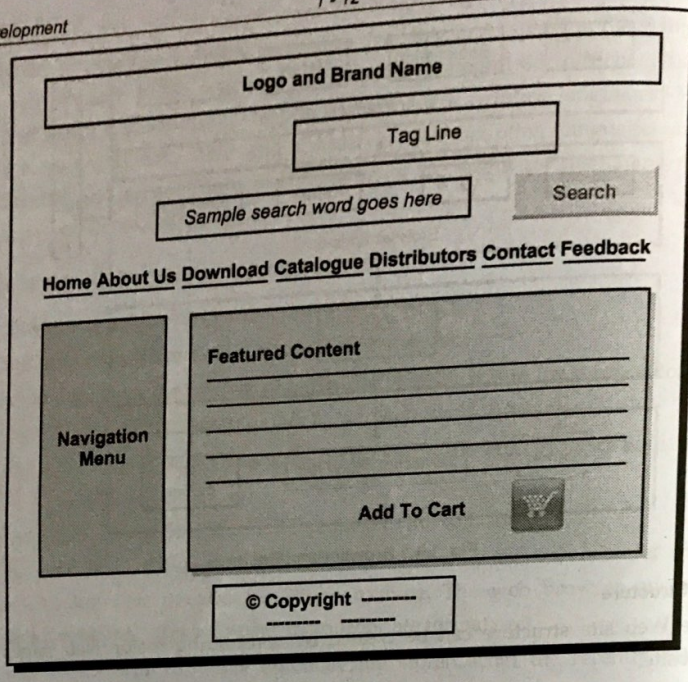


Fig. 1.4.2 Sample design sketch

1.5 Web Server Installation and Configuration

GTU : Summer-11,13

- Web server is a special type of server to which the web browser submits the request of web page which is desired by client.
- There are two popularly used web servers namely Apache and IIS server.

1.5.1 Web Server Operations

We often browse the internet for several reasons. It is more interesting to know about how a web page demanded by us gets displayed on our browser. Following is the stepwise explanation of this process -

Step 1 : First user types the websites address for demanding the desired web page for example -

http://www.vtubooks.com/home.aspx

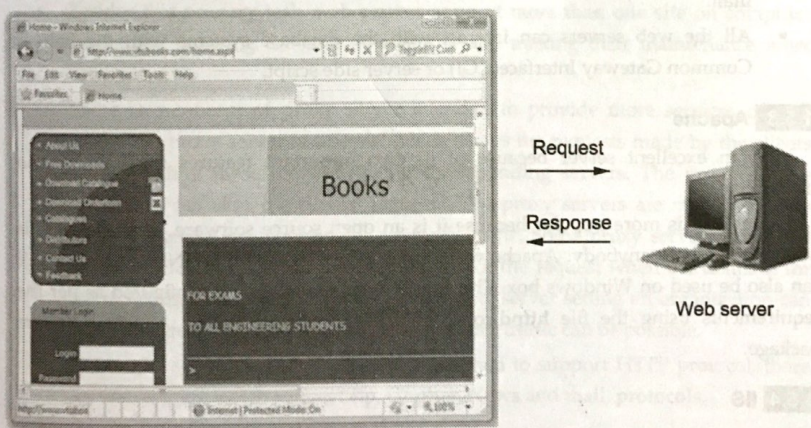
and then the home page of this website appears on the screen. The web address is divided into three parts: the first part is the protocol. The **http** is a hypertext

Scanned with CamScanner

transfer protocol which tells the web browser that user wishes to communicate with web server on **port 80**. Port 80 is reserved for the communication between web server and web browser. The second part is the server address. This tells the web browser which server it needs to contact in order to retrieve the information you are looking for. The web browser communicates with a Domain Name Server (DNS) to find out the IP Address for the website. All communications on the Internet use IP addresses for communications. Using the numeric address for accessing the web server is avoided because it is easier to remember textual information than that of numeric one. Hence normally the web server's addresses are textual. The third part of this address denotes the resource user wants to see.

Step 2 : The web browser, having found the IP address it needs by communicating with the name server, then sends a request directly to the web server, using port 80, asking for the file **home.aspx**.

Step 3 : The web server sends the html for this page back to user's web browser, which reads the HTML tags and formats them for viewing on your screen. If there are additional files needed in order to show the web page (like some images, for example) the web browser makes additional requests for each of these.



Web browser / Web client

Scanned with CamScanner

1.5.2 General Server Characteristics

- 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.

1.5.3 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.

1.5.4 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.

Scanned with CamScanner

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.

1.5.5 Functions of Web Server

- The primary task of web server is to **monitor communications** port on the host machine. It accepts the HTTP command using this port and performs the operation specified by the commands.
- Besides this primary task, web **servers support more than one site** on computer potentially reducing the cost of each site and making their maintenance more convenient.
- There is a concept of **proxy server** intended to provide more services of web server.The proxy server is a server which serves the requests made by the clients by forwarding those requests to the corresponding servers. The proxy servers sometimes can alter the clients' requests. The proxy servers are mostly used to control, monitor or unbound the network traffic. Some proxy servers **cache** the requested data. This helps in quick serving of the request when the demand for the previously read data is made. Using Proxy server setting an organization can block certain sites so that controlling of network traffic can be possible.
- Although web servers were originally designed to support HTTP protocol, there are web servers which support **ftp**, **Gopher**, **News** and **mail**, **protocols**.

Scanned with CamScanner

1.6 Web Security

There are some security principles that can be used to apply security mechanisms. These security principles are -

1. Authentication
2. Confidentiality
3. Integrity
4. Non-repudiation
5. Access Control
6. Availability

Let us discuss each of them in detail

1. Authentication

Authentication is a security mechanism which helps us to establish the proof of identities. That means the origin of the document can be correctly identified using authentication.

For example: If Mr.XYZ wants to transfer some money to Mr.PQR and if there comes Mr.LMN who pretends as if he himself is MR.PQR then money will get transferred from Mr.XYZ's account to Mr.LMN and not to actual Mr.PQR's account. This will happen due to lack of authentication mechanism. Such type of concept is also called as **fabrication**. Hence **fabrication** can be defined as a kind of attack that is possible when there is no authentication.

Scanned with CamScanner

2. Confidentiality

Another name to confidentiality is **secrecy**. That means the information sent by the sender should be read by the intended receiver and should not be read by any third party person. The kind of attack due to lack of confidentiality is called **interception**.

3. Integrity

Integrity is a kind of security in which the contents sent by the sender gets changed by some intruder and receiver gets the some different message.

For example - If Mrs. XYZ signs the cheque of ₹ 1000 and transfer it to Mr.PQR and if Mr.PQR gets an amount of ₹ 10000 then that means contents are modified in between. The attack in which the integrity of message gets lost is called **modification**.

4. Access Control

Access control is a mechanism by which it is determined which contents are accessible to the user and which are not. For example - User may be allowed to view the records from the database but is not allowed to update them.

5. Non-repudiation

When sender sends some message and later on refuses that he has sent that message then in such a situation there should some mechanism which will not allow the sender to refuse his sending. Such security mechanism is called **non-repudiation**.

For example : If Mr.XYZ has demanded for the transaction of Rs. 1000 and later on if he refuses about his claim then using non-repudiation mechanism bank can prove falsehoodness of Mr.XYZ.

6. Availability

Availability is the security principle by which the contents or the resources will be available to the authentic user for all the time. If some resources are demanded by the authentic user and will not be available to him due to some attack then such attack is called **interruption**

1.7 CORS

- Cross-Origin Resource Sharing (CORS) is a browser mechanism which allows to share the resources or data present on other site.
- Generally access to the resources that are residing in a third party site is restricted by the client side browser for security purpose.

Scanned with CamScanner

Identifying CORS Response Type

- When a server has been configured correctly to allow cross-origin resource sharing some special headers will be included. Their presence can be used to determine that a request supports CORS.
- Web browsers can use these headers to determine whether or not to process the request or not.
- Typically the header associated with the response is **Access-Control-Allow-origin**.
- For example, to allow access from any origin, you we can set this header as follows

```
Access-Control-Allow-Origin: *
```

- Or We can specify the specific origin

```
Access-Control-Allow-Origin: https://technicalpublications.com
```

Understanding CORS Request Type

- There are two types of CORS Requests -
  - Simple Requests
  - Preflight Requests
- The browser can determine which type of CORS request it is.
- For the simple requests -
  - The method for simple requests the method can be GET, POST or HEAD.
  - The CORS safe-listed headed is used.
  - The content type is normally application/x-www-form-urlencoded, ultipart/form-data, or text/plain.
- For Preflight requests -
  - The method OPTION is used.
  - The headers used in preflight requests are -
    - Access-Control-Request-Method: The intended method of the request (e.g. GET or POST)
    - Access-Control-Request-Headers: An indication of the custom headers that will be sent with the request
    - Origin: The usual origin header that contains the script's current origin

Review Question

1. *What is CORS? Explain CORS request types.*

Scanned with CamScanner

1.8 Understanding SEO

SEO stands for Search Engine Optimization. It is the practice of increasing quantity and quality of traffic to your web site through organic search engine result. Here organic search means a search for which you need not have to pay.

SEO is important because it is the practice of optimizing your web pages to make them reach a high position in the search results of Google or other search engines.

What are the benefits of SEO?

(1) **SEO Encourages Local Users to Visit the Physical Store After the Search:** For example – if an internet user searches for "Best restaurants in Pune" , then internet shows variety of options. The user will normally prefer the top choices and can become your customer if your restaurant has occupied higher position on the search engine page.

(2) **SEO builds the brand credibility:** Ranking first, second or third may give your customers the idea that you are one of the top players in the industry. It shows that you are popular and many users have researched you, too.

(3) **SEO helps establish brand awareness:** Brand Awareness is the extent to which the target market recognizes a brand. This refers to how familiar your customers are with your product or service.

(4) **SEO helps you gain market share:** Being on top of the search list means a high tendency for your website to be found by Internet users. These users are now considered as your leads. Once they have found their required information from your website, these leads may turn into your customers.

(5) **SEO helps you to create all marketing activities online:** All of your marketing strategies conducted online will contribute to the success of your search engine optimization efforts. Marketing activities such as content marketing, direct e-mail, social media marketing, blogging, web management, e-commerce, and others will help you get better rankings on several search engine sites.

How SEO Works?

- SEO works by demonstrating the search engines that your contents is the best result of user queries.
- Search Engine Optimization technique is mainly for optimizing the Google search Engine. Google uses more than 200 ranking factors. It is a good idea to have a knowledge about most commonly used ranking factors. Let us discuss some of them

Scanned with CamScanner

(1) Crawlability

- Google uses several ways to discover new content on the web, but the primary method is **crawling.** Basically crawling is where Google follows links on the pages. To do this, they use a computer program called a **spider.**
- Let us say that your homepage has a backlink from a website that is already in Google's index. Next time they crawl that site, they will follow that link to discover your website's homepage and likely add it to their index.
- From there, they will crawl the links on your homepage to find other pages on your site.
- Hence to adopt SEO, Google's crawler should not get blocked because of . i) poor internal ranking ii) no properly followed internal links among web pages of your website iii) no index pages.

(2) Mobile Friendliness

Most of the searches to Google come from mobile devices. Hence Google has started mobile version of your page for indexing and ranking. Pages that aren't optimized for mobile lead to dissatisfaction. Hence it is necessary to have your web pages being mobile friendly.

(3) Page Speed

Page speed means how fast your page loads. It is a ranking factor on desktop and mobile. If the user clicks on your page displayed as a search result and if it takes too long to load, then that certainly leads to dissatisfaction. Therefore it is necessary to check the page speed of your web site and keep it improving.

(4) Content Quality

- Google wants to rank the most reliable and useful results - always.
- To do this, they look at content-related signals like expertise, authoritativeness, and trustworthiness.
- To adopt SEO the quality of the contents of your web page must be full proof.

Review Questions

1. *What is SEO ? What are the benefits of it ?*
2. *Explain the working of SEO.*

Scanned with CamScanner

3.1 Features of JavaScript

Following are some features of JavaScript

1. **Browser Support :** For running the JavaScript in the browser there is no need to use some plug-in. Almost all the popular browsers support JavaScripting.
2. **Structure Programming Syntax :** The Javascript supports much commonly the structured language type syntax. Similar to C-style the programming blocks can be written.
3. It automatically inserts the semicolon at the end of the statement, hence there is no need to write semicolon at the end of the statement in JavaScript.
4. **Dynamic Typing :** It supports dynamic typing, that means the data type is bound to the value and not to the variable. For example one can assign integer value to a variable say 'a' and later on we can assign some string value to the same variable in JavaScript.
5. **Run Time Evaluation :** Using the **eval** function the expression can be evaluated at run time.
6. **Support for Object :** JavaScript is object oriented scripting language. However handling of objects in JavaScript is somewhat different that the conventional object oriented programming languages. JavaScript has a small number of in-built objects.
7. **Regular Expression :** JavaScript supports use of regular expressions using which the text-pattern matching can be done. This feature can be used to validate the data on the web page before submitting it to the server.
8. **Function Programming :** In JavaScript functions are used. One function can accept another function as a parameter. Even, one function can be assigned to a variable just like some data type. The function can be run without giving the name.

3.2 JavaScript Syntax and Types

The JavaScript can be directly embedded within the HTML document or it can be stored as external file.

Directly embedded JavaScript

The syntax of directly embedding the JavaScript in the HTML is

```
<script type="text/javascript">
...
```


Scanned with CamScanner

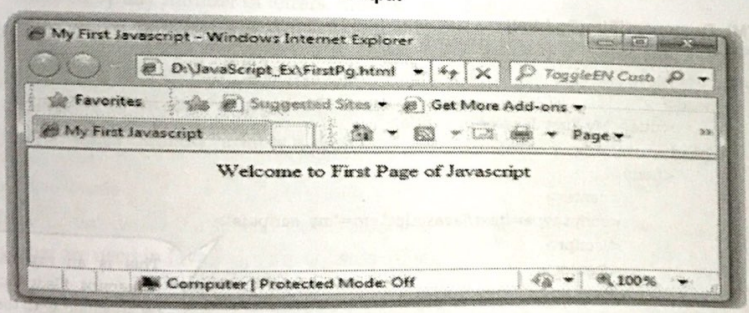
```
        ...
        ...
</script>
```

Example 3.2.1 *Write a JavaScript to display Welcome message in JavaScript*

Solution :

```
<!DOCTYPE html >
<html >
   <head>
      <title> My First Javascript </title>
   </head>
   <body>
      <center>
      <script type="text/javascript">
      /*This is the First JavaScript*/
      document.write(" Welcome to First Page of Javascript");
      </script>
      </center>
   </body>
</html>
```

Output



Script Explanation :

In above script

(1) We have embedded the javaScript within

```
   <script type="text/javascript">
   ...
   </script>
```

Scanned with CamScanner

(2) A comment statement using /* and */. Note that this type of comment will be recognized only within the <script> tag. Because, JavaScript supports this kind of comment statement and not the XHTML document.

(3) Then we have written document.write statement, using which we can display the desired message on the web browser.

Indirectly Embedding JavaScript

If we want to embed the JavaScript indirectly, that means if the script is written in some another file and we want to embed that script in the HTML document then we must write the script tag as follows -

```
<script type="text/javascript" src="MyPage.js">
    ...
    ...
    ...
</script>
```

Javascript is which is to be embedded is in the file MyPage.js

We will follow the following steps to use the external JavaScript file.

Step 1 : Create an XHTML document as follows -

XHTML Document[FirstPg.html]

```
<!DOCTYPE html >
<html>
    <head>
        <title> My First Javascript </title>
    </head>
        <body>
            <center>
            <script type="text/javascript" src="my_script.js">
            </script>
            </center>
        </body>
    </html>
```

This is an external javascript file,it can be specified with the attribute **src**

Step 2 :

JavaScript[my_script.js]

```
/*This is the First JavaScript*/
        document.write(" Welcome to First Page of Javascript");
```

Step 3 : Open the HTML document in Internet Explorer and same above mentioned output can be obtained.

Scanned with CamScanner

Advantages of indirectly embedding of JavaScript

1. Script can be hidden from the browser user.
2. The layout and presentation of web document can be separated out from the user interaction through the JavaScript.

Disadvantages of indirectly embedding of JavaScript

1. If small amount of JavaScript code has to be embedded in XHTML document then making a separate JavaScript file is meaningless.
2. If the JavaScript code has to be embedded at several places in XHTML document then it is complicated to make separate JavaScript file and each time invoking the code for it from the XHTML document.

3.3 Keywords, Identifiers, and Comments

1. Identifiers

Identifiers are the names given to the variables. These variables hold the data value. Following are some conventions used in JavaScript for handling the identifiers -

1. Identifiers must begin with either letter or underscore or dollar sign. It is then followed by any number of letters, underscores, dollars or digits.
2. There is no limit on the length of identifiers.
3. The letters in the identifiers are case-sensitive. That means the identifier INDEX, Index, index, inDex are considered to be distinct.
4. Programmer defined variable names must not have upper case letters.

2. Reserved words

Reserved words are the special words associated with some meaning. Various reserve words that are used in JavaScript are enlisted as below –

break	Continue	delete	for	in	return	throw	var	with
case	default	else	function	instanceof	switch	try	void	
catch	do	finally	if	new	this	typeof	while	

3. Comments

JavaScript supports following comments

1. The // i.e a single line comment can be used in JavaScript.
2. The /* and */ can be used as a multi-line comment.

Scanned with CamScanner

3. The XHTML <!--> and <--> is also allowed in JavaScript

4. Semicolon

While writing the JavaScript the web programmer must give semicolon at the end of the statements

3.4 Data Types

JavaScript defines two entities primitives and objects. The primitives are for storing the values whereas the object is for storing the reference to the actual value.

There are following primitive types used in JavaScript

1. Number 2. String 3. Boolean
4. Undefined 5. Null

There are three type of predefined objects in JavaScript

1. Number 2. String 3. Boolean

These objects are called **wrapper objects**. These wrapper objects provide properties and methods which can be used by primitive types.

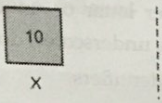


Fig. 3.4.1 (a) Representation of primitive type

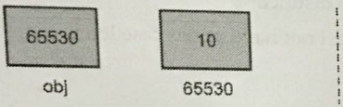


Fig. 3.4.1 (b) Representation of object

3.5 Variables

In JavaScript we can declare the variable using the reserved word var. The value of this variable can be any thing; it can be numeric or it can be string or it can be a Boolean value.

JavaScript[VarDemo.html]

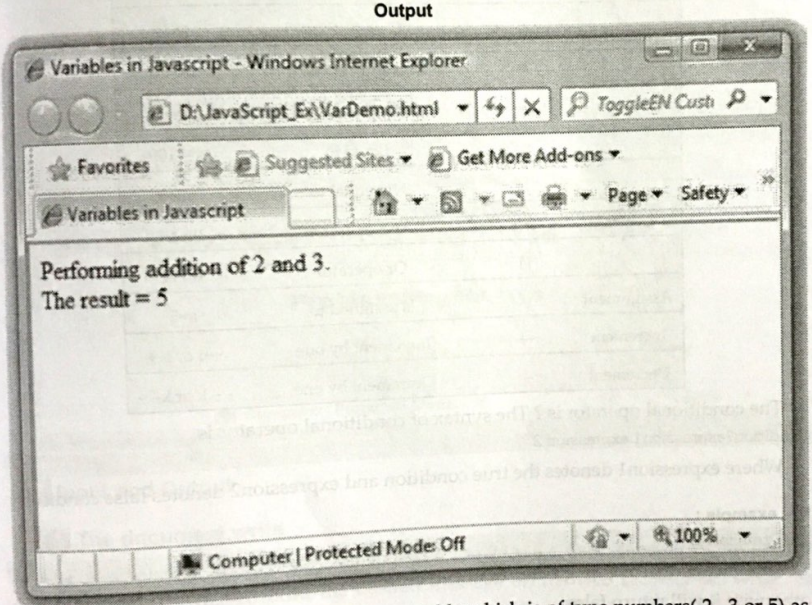
```
<!DOCTYPE html >
<html >
 <head>
 <title> Variables in Javascript </title>
</head>
```

Scanned with CamScanner

```
<body>
<script type="text/javascript">
var a,b,c;
var string;
a=2;
b=3;
c=a+b;
string ="The result = ";
document.write("Performing addition of 2 and 3.  "+"<br/>");
document.write(string);
document.write(c);
</script>
</body>
</html>
```

Variable declaration is done using **var**. Note that there is no data type required for handling variables.

Output



Note that using **var** we can define the variable which is of type numbers(2 , 3 or 5) as well as the string "The result".

Scanned with CamScanner

3.6 Operators

Various operators used by JavaScript are as shown in following table -

Type	Operator	Meaning	Example
Arithmetic	+	Addition or unary plus	c = a+b
	-	Subtraction or unary minus	d = -a
	*	Multiplication	c=a*b
	/	Division	c=a/b
	%	Mod	c=a%b
Relational	<	Less than	a<4
	>	Greater than	b>10
	<=	Less than equal to	b<=10
	>=	Greater than equal to	a>=5
	==	Equal to	x==100
	!=	Not equal to	m!=8
Logical	&&	And operator	0&&1
	||	Or operator	0 ||1
Assignment	=	Is assigned to	a=5
Increment	++	Increment by one	++i or i++
Decrement	--	Decrement by one	--k or k--

The conditional operator is ? The syntax of conditional operator is

```
Condition?expression1:expression 2
```

Where expression1 denotes the true condition and expression2 denotes false condition.

For example :

```
a>b?true:false
```

This means that if a is greater than b then the expression will return the value true otherwise it will return false.

3.6.1 String Concatenation Operator

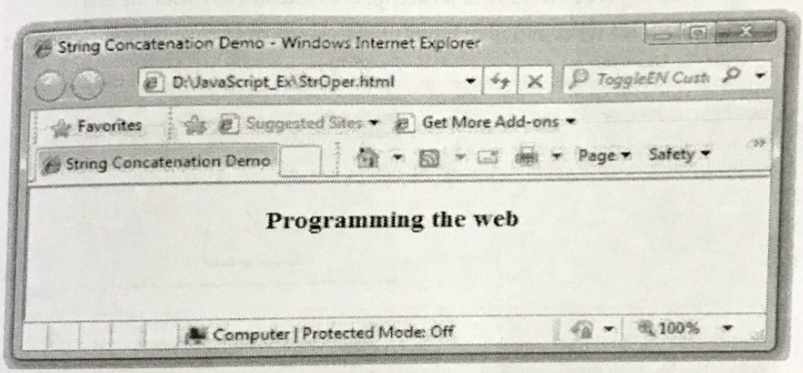
Two string can be concatenated using the + operator. A variable can be concatenated with the string using + operator also.

Scanned with CamScanner

JavaScript[StrOper.html]

```
<!DOCTYPE html >
<html >
<head>
 <title>String Concatenation Demo</title>
</head>
<body>
 <center>
 <script type="text/javascript">
  var first_string;
  first_string="Programming";
  document.write("<h3>"+first_string+" the web"+"</h3>");
 </script>
 </center>
</body>
</html>
```

Output



3.7 Input and Output

3.7.1 The document.write

- For displaying the message on the web browser the basic method being used is **document.write**. To print the desired message on the web browser we write the message in double quotes inside the document.write method.
- **For example**

```
document.write("Great India");
```

- We can pass the variable instead of a string as a parameter to the document.write.

Scanned with CamScanner

For example

```
var my_msg="Great India";
document.write(my_msg);
```

Note that the variable my_msg should not be passed in double quotes document.write otherwise the result the string "my_msg" will be printed on the browse instead of the string "Great India" .

- We can combine some HTML tags with the variable names in document.write s that the formatted display can be possible on the web browser.

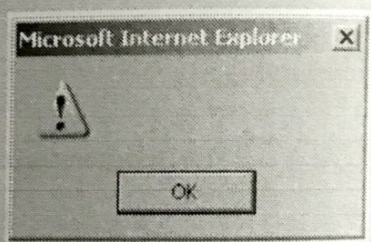
For example - if we want to print the message "Great India" on the web browser i bold font then we can write following statements-

```
var my_msg="Great_India";
document.write("<strong>"+my_msg+"</strong>");
```

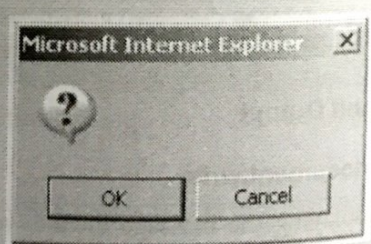
3.7.2 Popup Box

- One of the important features of JavaScript is its **interactivity** with the user.
- There are **three types of popup boxes** used in JavaScript by which user can interac with the browser.

alert box : In this type of popup box some message will be displayed.

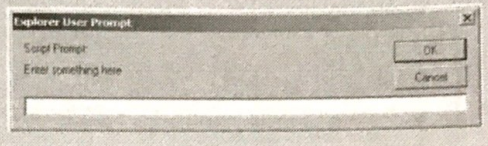


confirm box: In this type of popup box in which the message about confirmation will be displayed. Hence it should have two buttons **Ok** and **Cancel.**



Scanned with CamScanner

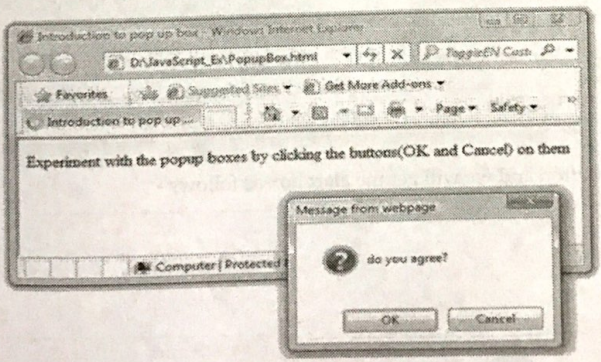
Prompt box is a type of popup box which displays a text window in which the user can enter something. Hence it has two buttons **Ok** and **Cancel**.



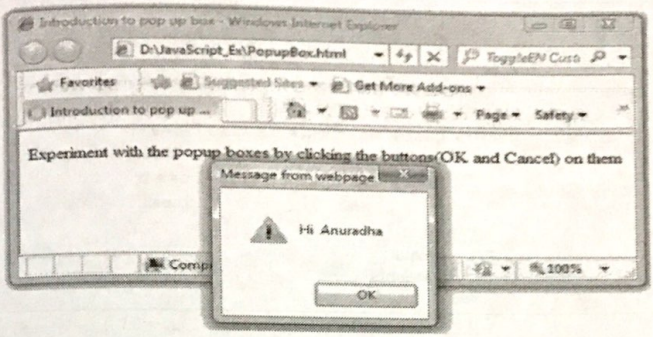
JavaScript[PopupBox.html]

```
<!DOCTYPE html>
<html >
<head>
  <title>Introduction to pop up box</title>
</head>
<body>
<p>Experiment with the popup boxes by clicking the buttons(OK and Cancel) on them</p>

  <script type="text/javascript">
   if(confirm("do you agree?"))
     alert("You have agreed");
   else
     input_text=prompt("Enter some string here...","");
   /*the value entered in prompt box is returned
   and stored in the variable text */
     alert("Hi "+input_text);
  </script>
</body>
</html>
```

Output



Scanned with CamScanner



Thus alert, confirm and prompt boxes cause the browser to wait for user response. The user responds by clicking the button on these popup boxes.

Script Explanation

On loading this script using web browser first of all a confirm box will be displayed. If we click on OK button an alert box will appear. Otherwise a prompt box will be displayed. Again if we click on the OK button of prompt box again an alert box will appear which will display the string which you have entered on prompt box. If you run the above script you will get all these effects.

3.8 Conditions and Loops

Various control structures used in JavaScript are

Statement	Syntax	Example

if-else	if(condition)
	statement
	else
	statement
		if(a>b)
		document.write(" a is greater than b");
		else
		document.write("b is greater than a");

while	while(condition)
	{
	statements
	}
		while(i<5)
		{
		i=i+1;
		document.write("value of i"+i)
		}

do..while	do
	{
	}while(condition);
		do
		{
		i=i+1;
		document.write("value of i"+i)
		}while(i<5);

for	for(initialization;test condition;stepcount)	for(i=0;i<5;i++) {

Scanned with CamScanner

	{
	statements
	}
		document.write(i);
		}

switch...case	switch(expression)
	{
	case 1: statements
	break;
	case 2: statements
	break;
	...
	default:
	statements
	}
		switch(choice)
		{
		case 1: c=a+b;
		break;
		case 2:c=a-b;
		break;
		}

break :
Similar to C or C++, the break statement is used to break the loop.
	break;	for(i=10;i>=0;i--)
		{
		if(i==5)
		break;
		}

continue:
The continue statement is used in a loop in order to continue(skip). The keyword **continue** is used to make use of continue statement in a loop.
	continue;	for(i=10;i>=0;i--)
		{
		if(i==5)
		{
		x=i;
		continue;
		}

Control structure is essential part of JavaScript. The Control Structures in JavaScript are just similar to that of C or C++. Various control structures are -

1. if statement 2. While 3. Do-while 4. for
5. switch case 6. break 7. Continue

Let us discuss various examples that make use of control structures.

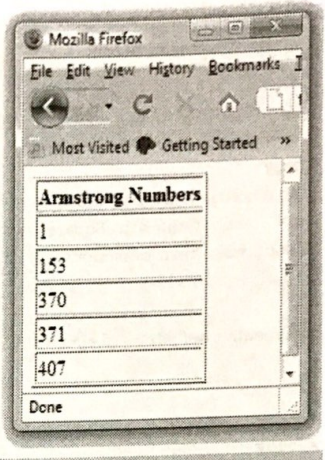
Example 3.8.1 *Develop a javascript to generate 'ARMSTRONG NUMBERS' between the range 1 to 100. [Eg : 153 is an Armstrong number, since sum of the cube of the digits is equal to the number i.e. [13+53+33=153]*

Solution :

```
<html>
<body>
<table border="1" allign="center">
<th>Armstrong Numbers</th>
<script type="text/javascript">
var num,i,temp,sum;
```

Scanned with CamScanner

```
var n=0;
i=1;
do
{
 num=i;
 sum=0;
 while(num>0)
 {
 n=num%10;
 n=parseInt(n);
 num=num/10;
 num=parseInt(num);
 sum=sum+(n*n*n);
 }
 if(sum==i)
 {
 document.write("<tr><td>"+i+"</td></tr>");
 }
 i++;
}while(i<=1000);
</script>
</table>
</body>
</html>
```



Example 3.8.2 *Write a javascript that displays as per following :*

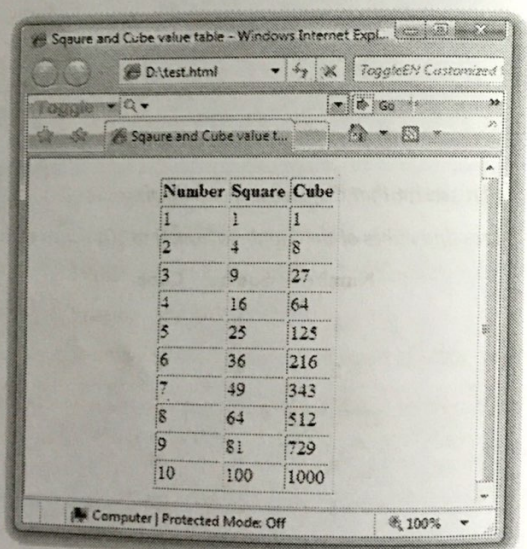
(Calculate the squares and cubes of the numbers from 0 to 10)

Number	Square	Cube
1	1	1
2	4	8
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

Scanned with CamScanner

Solution :

```
<html>
<head>
<title>Sqaure and Cube value table</title>
</head>
<body>
<table border=1 align="center">
<th>Number</th><th>Square</th><th>Cube</th>
<script type="text/javascript">
for (i=1; i<=10; i++)
{
document.write("<tr><td>"+i+"</td><td>"+(i*i)+"</td><td>"+(i*i*i)+"</td></tr>");
}
</script>
</table>
</body>
</html>
```



Number	Square	Cube
1	1	1
2	4	8
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

Example 3.8.3 *Write a script that reads an integer and displays whether it is a prime number or not.*

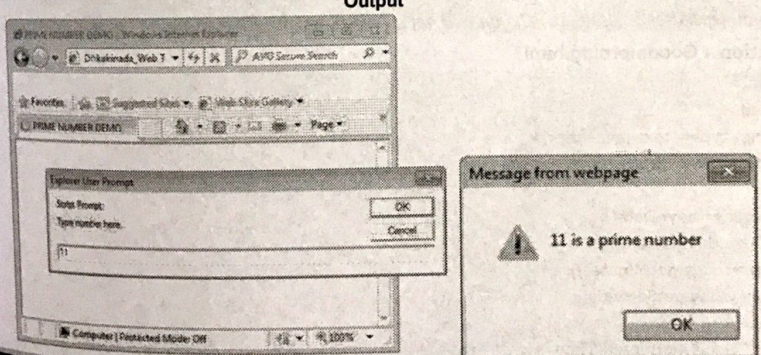
Solution :

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

Scanned with CamScanner

```
<title>PRIME NUMBER DEMO</title>
</head>
<body>
<script type="text/javascript">
var num=prompt("Type number here.","");
var b;
var flag=1;
for(i=2;i<num;i++)
{
  b=num%i;
  if(b==0)
  {
   flag=0;

   break;
  }
}
if(flag==0)
  alert(num+" is not a prime number");
else
  alert(num+" is a prime number");
</script>
</body>
</html>
```

Output



Example 3.8.4 *Write a JAVAScript to print characters of a string at odd positions. (for example for the string India, I, d and a should get printed).*

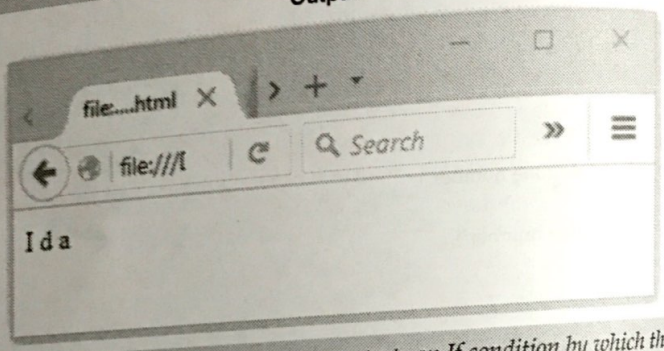
Solution :

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

Scanned with CamScanner

```
<script>
var str="India";
var k=str.length;
for(i=0;i<=k;i=i+2)
{
n=str.charAt(i);
document.write(n);
document.write(" ");
}
</script>
</body>
</html>
```

Output



Example 3.8.5 *Develop a JavaScript page to demonstrate an If condition by which the time on your browser is less than 10; you will get a "Good morning" greeting.*

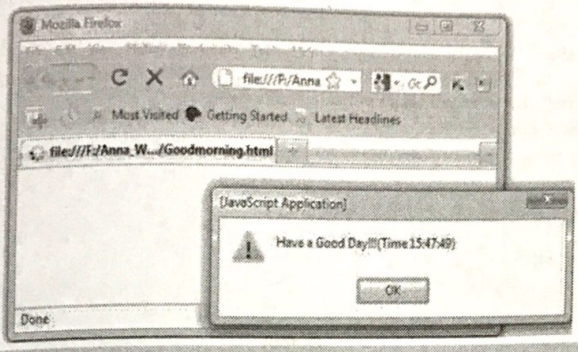
Solution : Goodmorning.html

```
<html>
<head>
<script type="text/javascript">
function GreetMsg()
{
 var today=new Date();
 var h=today.getHours();
 var m=today.getMinutes();
 var s=today.getSeconds();
 if(h<10)
  alert("Good Morning!!!(Time "+h+":"+m+":"+s+")");
 else
  alert("Have a Good Day!!!(Time "+h+":"+m+":"+s+")");
}
</script>
</head>
<body onload="GreetMsg()">
```

Scanned with CamScanner

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

Output



Example 3.8.6 *Write a JavaScript to find and print the largest and smallest values among 10 elements of an array.*

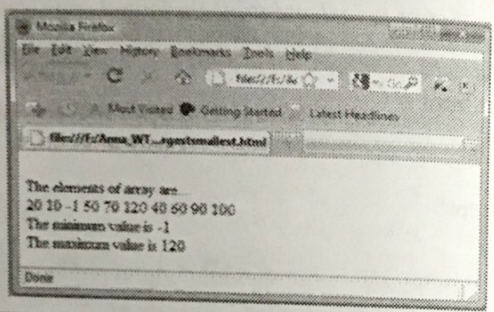
Solution : largestsmallest.html

```
<html>
 <head>
 <script type="text/javascript">
 function fun()
 {
  a=new Array(10);
  a[0]=20;
  a[1]=10;
  a[2]=-1;
  a[3]=50;
  a[4]=70;
  a[5]=120;
  a[6]=40;
  a[7]=60;
  a[8]=90;
  a[9]=100;
  document.write("<br/>The elements of array are...<br/>");
  max_val=a[0];
  for(i=0;i<10;i++)
   document.write(" "+a[i]);
  min_val=a[0];
  max_val=a[0];
  for(i=0;i<10;i++)
```

Scanned with CamScanner

```
 {
   if(a[i]<min_val)
     min_val=a[i];
   if(a[i]>max_val)
     max_val=a[i];
 }
 document.write("<br/>The minimum value is "+min_val);
 document.write("<br/>The maximum value is "+max_val);
}
 </script>
<title>Finding largest and smallest Element</title>
</head>
 <body onload=fun()>
 </body>
</html>
```

Output



Example 3.8.7 *Write a JavaScript to find the product of first 15 even numbers.*

Solution :

```
<html>
<body>
<script type="text/javascript">
prod=1;
for(i=1;i<15;i++)
{
    if(i%2)
    {
        prod=prod*i;
    }
}
document.write("Product of odd numbers between 1 to 15 is "+prod);
</script>
</body>
</html>
```


Scanned with CamScanner

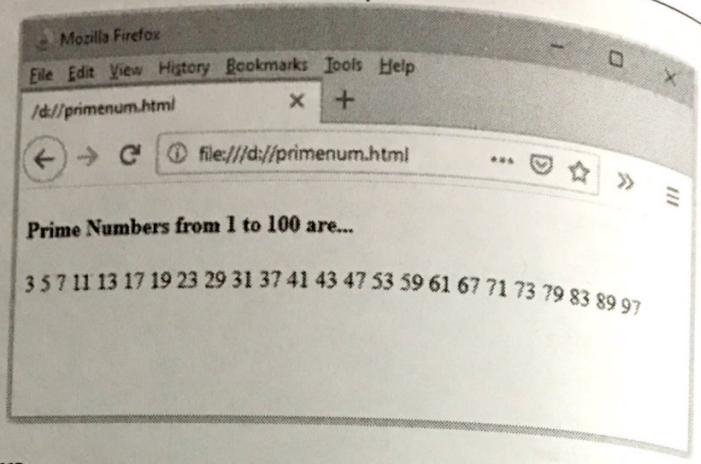
Example 3.8.8 *Write a JavaScript program to print prime numbers from 1 to 100.*

Solution :

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

function primeNumber(from, to){
        var flag=false;
        for(i = from; i <= to; i++)
       {
        for( j = 2; j < i; j++)
        {
        if(i % j == 0)
            {
                flag = false;
                break;
            }
        else
            {
                flag = true;
            }
        }
        if(flag)
        {
        document.write(" "+i);
        }
      }
    }
</script>
</head>
<body>
<h4> Prime Numbers from 1 to 100 are...</h4>
<script type="text/javascript">
primeNumber(1, 100)
</script>
</body>
</html>
```

Scanned with CamScanner

Output



3.9 Arrays

- Arrays is a collection of similar type of elements which can be referred by a common name.
- Any element in an array is referred by an array name followed by "[" followed by position of the element followed by].
- The particular position of element in an array is called array index or subscript.

For example –

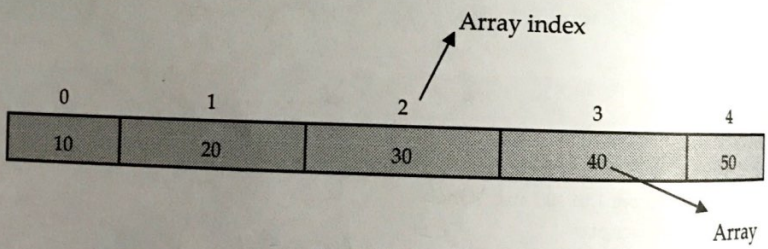


Fig. 3.9.1 Arrays

- Normally the first element in an array is stored at 0th location, however we can start storing the element from any position.

3.9.1 Array Declaration

- In JavaScript the array can be created using **Array** object.
- Suppose, we want to create an array of 10 elements then we can write,

Scanned with CamScanner

```
var ar = new Array(10);
```

- Using new operator we can allocate the memory dynamically for the arrays.
- In the brackets the size of an array is mentioned and the var ar denotes the name of the array. Thus by the above sentence an array ar will be created in which we can store 10 elements at the most. Sometimes the above statement can be written like this

```
var ar;
   ar=new Array(10);
```

3.9.2 Array Initialization

Let us see how to store some elements in an array.

- **JavaScript Program [ArrayDemo.html]**

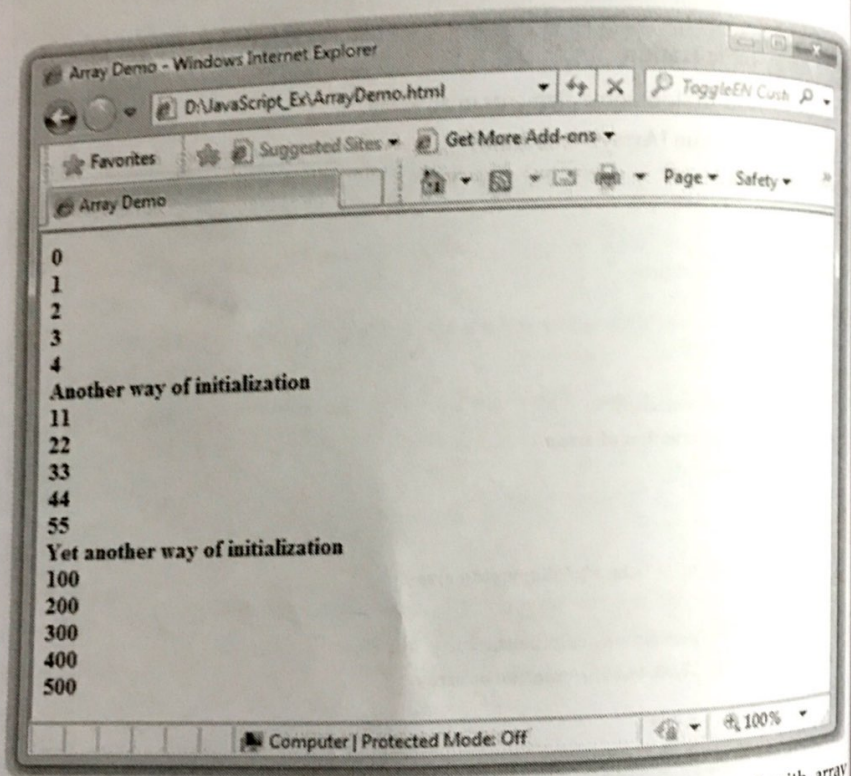
```
<!DOCTYPE html>
<html>
<head>
  <title>Array Demo</title>
</head>
<body>
<strong>
  <script type="text/javascript">
   a=new Array(5);//creation of array
   for(i=0;i<5;i++)
   {
    a[i]=i;
    document.write(a[i]+"<br>");//displaying array
   }
   document.write("Another way of initialization"+"<br>");
   b=new Array(11,22,33,44,55);//creation of array
   for(i=0;i<5;i++)
   {
    document.write(b[i]+"<br>");//displaying array
   }
   document.write("Yet another way of initialization"+"<br>");
   var c=[100,200,300,400,500];//creation of array
   for(i=0;i<5;i++)
   {
    document.write(c[i]+"<br>");//displaying array
```


Scanned with CamScanner

```
    }
</script>
</strong>
</body>
</html>
```

As you can notice that, in above JavaScript an array can be initialized in three differen ways which is shown by boldface. Hence an output of above script will be

Output



There is one control structure in JavaScript which is closely associated with array elements and such a control structure is **for...in**. Let us see a simple JavaScript which makes use of **for-in** control structure to display elements of an array.

Scanned with CamScanner

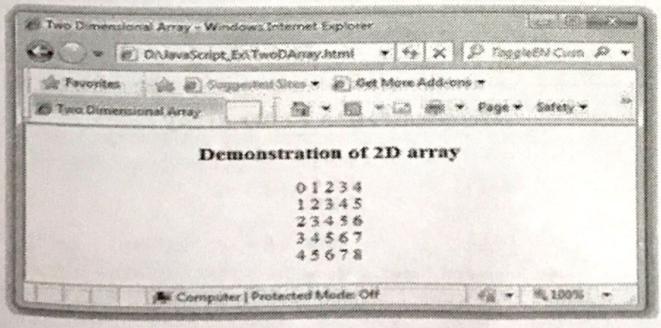
3.9.3 Two Dimensional Array

Actually JavaScript does not support the multidimensional arrays. Hence for defining the 2D array we make use of single dimensional array, how? Here it is -

JavaScript[TwoDArray.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Two Dimensional Array</title>
</head>
<body>
 <center>
 <h3> Demonstration of 2D array</h3>
 <script type="text/javascript">
  a=new Array();//creation of rows of array
  for(i=0;i<5;i++)
  a[i]=new Array();//creating columns of array
  for(i=0;i<5;i++)
  {
   for(j=0;j<5;j++)
   {
    a[i][j]=i+j;
    document.write(a[i][j] +"  ");
   }
   document.write("<br>");
  }
 </script>
</center>
</body>
</html>
```

Output



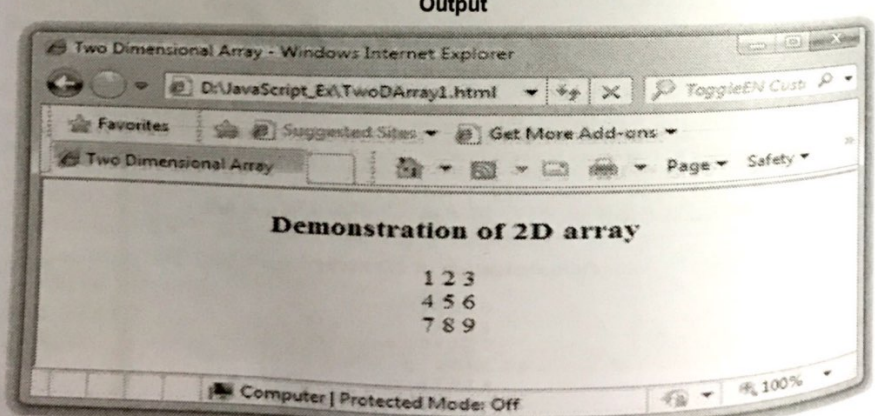
Scanned with CamScanner

There is another way by which the two dimensional array can be initialized. This method is represented in the following Script

JavaScript[TwoDArray1.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Two Dimensional Array</title>
</head>
<body>
 <center>
 <h3> Demonstration of 2D array</h3>
 <script type="text/javascript">
  var a=[ [1,2,3],
          [4,5,6],
          [7,8,9]
        ]; //creation of array
  for(i=0;i<3;i++)
  {
  for(j=0;j<3;j++)
  {
  document.write(a[i][j] +"  ");
  }
  document.write("<br>");
  }
 </script>
</center>
</body>
</html>
```

Output



Scanned with CamScanner

```
sum=0;
for(i=1;i<n;i++)
{
if(i%2==0)
  sum=sum+i;
}
document.write("<br>"+" Sum of first "+n+" even numbers is "+sum+"<br>");
}
</script>
</head>
<body>
<script type="text/javascript">
  var n=prompt("Enter the value of n","");
  EvenNumSum(n);
</script>
</body>
</html>
```

3.11 JavaScript Objects and DOM

3.11.1 Definition of DOM

- The Document Object Modeling (DOM) is for defining the standard for accessing and manipulating HTML,XML and other scripting languages.
- It is the **W3C recommendation** for handling the structured documents.
- Normally the structured information is provided in XML, HTML and many other documents.
- Hence DOM provides the standard set of programming interfaces for working with XML and XHTML and JavaScript.

What is DOM?

Document Object Model (DOM) is a set of platform independent and language neutral Application Programming Interface (API) which describes how to access and manipulate the information stored in XML,HTML and JavaScript documents.

3.11.2 DOM Tree

- Basically DOM is an **Application Programming Interface (API)** that defines the interface between HTML document and application program. That means, suppose application program is written in Java and this Java program wants to access the

Scanned with CamScanner

elements of HTML 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.
- 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>
    </html>
```

- The DOM tree will be

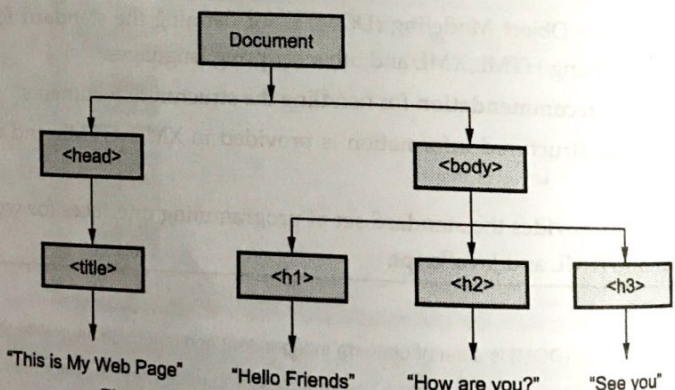


Fig. 3.11.1 DOM structure for simple web document

- We can describe some basic terminologies used in DOM tree as follows -
  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.**

Scanned with CamScanner

3.11.3 Using DOM Methods

We can access or change the contents of document by using various methods. Some commonly used properties and methods of DOM are as follows :

Methods

Method	Meaning
getElementById	This method is used to obtain the specific element which is specified by some id within the script.
createElement	This method is used to create an element node.
createTextNode	Useful for creating a text node.
createAttribute	Useful for creating attribute.
appendChild	For adding a new child to specified node, this method is used.
removeChild	For removing a child node of a specific node, this method is used.
getAttribute	This method is useful for returning the specified attribute value.
setAttribute	This method is useful for setting or changing the specified attribute to the specified value.

Properties

Property	Meaning
attributes	This property is used to get the attribute nodes of the node.
parentNode	This DOM property is useful for obtaining the parent node of the specific node.
childNodes	This DOM property is useful for obtaining the child nodes of the specific node.
innerHTML	It is useful for getting the text value of a node.

3.11.3.1 Accessing Elements using DOM

- 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.

```
<html >
<head>
   <title>This is My Web Page </title>
</head>
<body>
```

Scanned with CamScanner

```
<form name="form1">
    <input type="text" name="myinput"/>
</form>
</body>
</html>
```

Method 1

- Every XHTML document element is associated with some address. This address called **DOM address.**
- The document has the collection of **forms** and **elements**. Hence we can refer the te box element as

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

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

```
...
<form name="form1">
    <input type="button" name= "mybutton"/>
    <input type="text" name="myinput"/>
</form>
...
```

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

Method 2

- We can access the desired element from the web document using JavaScript me **getElementById**. 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 elemer the form such as radio buttons or check boxes then they normally appear groups. Hence to access these elements we make use of its index. Consid 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++)
```

Scanned with CamScanner

```
  document.write(parseInt("100"));
 </script>
</body>
</html>
```

The output of above code will be 100.

Similar to the **parseInt** function the **parseFloat** function is used to obtain the real value from the string.

4. eval

The eval function is used to evaluate the expression.

The syntax is

```
eval(string);
```

Example

globlafun4.html

```
<html>
 <body>
  <script type="text/javascript">
   document.write(eval("2+3*5"));
  </script>
 </body>
</html>
```

The output of above code will be **17.**

3.13 Javascript Validations

- Various control objects are placed on the form. These control objects are called **widgets.**
- These widgets used in JavaScript are – Textbox, Push button, Radio button, Checkbox and so on.
- In JavaScript the **validation of these widgets** is an important task.

Let us understand the validation of form elements with the help of examples –

Example 3.13.1 *Write a JavaScript for password verification.*

Solution :

```
<!DOCTYPE html>
<html>
<head>
 <title>Demo of onlclick Tag Attibute</title>
<script type="text/javascript">
```

Scanned with CamScanner

```
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>
```

Scanned with CamScanner

```
Course:
<select name="course">
<option value="">Select an Option:</option>
<option value="Computer">Computer</option>
<option value="Mechanical">Mechanical</option>
<option value="E&Tc">E&Tc</option>
</select>
<br/><br/>
<input type="button" name="SubmitButton" value="Submit"
onClick="ValidateForm(this.form)">
<input type="reset" value="Reset">
</form>
</body>
</html>
```

3.4 Regular Expressions

- **Definition :** Regular Expression is a special text string that defines the search pattern. It is a logical expression.
- **For example** – For counting specific characters in a string or to replace some substring by another substring we need to create a regular expression.
- We can create a regular expression pattern using forward slash /. For instance -

  re = /abc/
- Regular expression is a powerful way for searching and replacing the characters in the string.
- The words of regular expression are called **special characters.**
- Various special characters that can be used in Regular expression along with their meanings are written in the following table :

Special Character	Meaning
.	Any character except newline
A	The character a
ab	The string ab
a|b	a or b
a*	0 or more a's
\	Escapes a special character
[ab-d]	One character of: a, b, c, d

Scanned with CamScanner

[^ab-d]	One character except: a, b, c, d
[\b]	Backspace character
\d	One digit
\D	One non-digit
\s	One whitespace
\S	One non-whitespace
\w	One word character
\W	One non-word character
*	0 or more
+	1 or more
?	0 or 1
{2}	Exactly 2
{2, 5}	Between 2 and 5
{2,}	2 or more
(...)	Group of pattern
^	Start of string
$	End of string
\b	Word boundary
\n	Newline
\r	Carriage return
\t	Tab
\0	Null character

Methods that use regular expressions

Method	Description
exec	A RegExp method that executes a search for a match in a string. It returns an array of information or null on a mismatch.

Scanned with CamScanner

test	A RegExp method that tests for a match in a string. It returns true or false.
match	A String method that executes a search for a match in a string. It returns an array of information or null on a mismatch.
matchAll	A String method that returns an iterator containing all of the matches, including capturing groups.
search	A String method that tests for a match in a string. It returns the index of the match, or –1 if the search fails.
replace	A String method that executes a search for a match in a string, and replaces the matched substring with a replacement substring.
split	A String method that uses a regular expression or a fixed string to break a string into an array of substrings.

3.14.1 Finding Non Matching Characters

We can find the non matching characters from the given text by placing ^ as the first character within a square [ ].

Example 3.14.1 *Write a Java program that checks whether the string entered by the user contains digit or not.*

Solution :

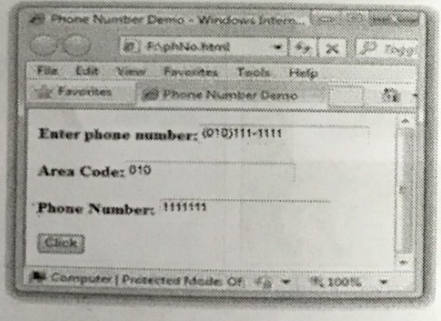
```
<!DOCTYPE html>
<html>
<head>
   <script type="text/javascript">
      function TestString(str)
      {
        re=/[^[0-9]]/; // [0-9] indicates any digit
        if(re.test(str))
         {
         alert("The string does not contain
```


Scanned with CamScanner

Solution : HTML Document[phNo.html]

```
<html>
<head>
  <title>Phone Number Demo</title>
  <script type="text/javascript">
  function TestString()
   {
   var str=document.form1.input.value;
   var i=str.match(/("\d{3}")"\" "\d{3}"-"\d{4}/);
   var temp1=str.split("(");
   var temp2=temp1[1].split(")")
   document.form1.area.value=temp2[0];
   var temp3=temp2[1].split(" ");
      var temp4=temp3[1].split("-")
   document.form1.phnum.value=temp4[0]+temp4[1];
   }
  </script>
</head>
 <body>
   <form name="form1">
   <b>Enter phone number:</b><input type="text" name="input" value=""><br/><br/>
   <b>Area Code:</b><input type="text" name="area" value=""><br/><br/>
   <b>Phone Number:</b> <input type="text" name="phnum" value=""><br/><br/>
   <input type="button" value="Click"onclick=TestString(input)>
 </body>
</html>
```

Output



3.15 Event Handling with Javascript

- **Definition of Event :** Event is an activity that represents a change in the environment. For example mouse clicks, pressing a particular key of keyboard represent the events. Such events are called intrinsic events.

Scanned with CamScanner

- **Definition of Event Handler :** 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.
- **Event Registration :** 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.

Internet Programming

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

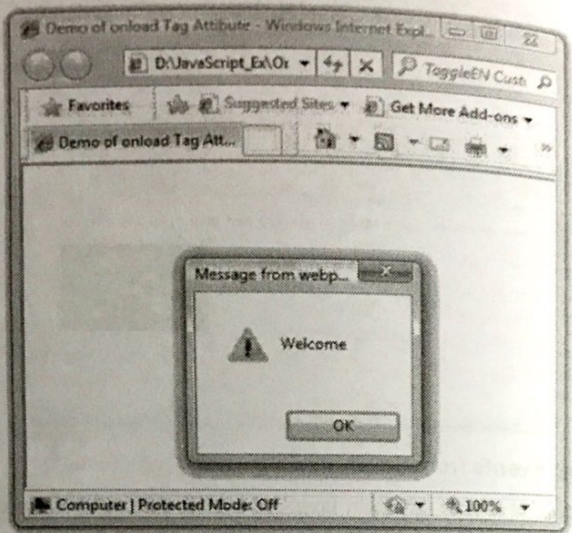
Commonly used events and tag attributes are enlisted in the following table -

Events	Intrinsic event 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>

Scanned with CamScanner

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

Output



3.16 Callbacks in Javascript

In JavaScript functions are objects. We can pass objects to the function as a parameter. That means we can pass function as a parameter to another function.

The mechanism of passing function as a parameter to another function is called callbacks. For instance –

```
function print(callback) {
  callback();
}
```

The print( ) function takes another function as a parameter and calls it inside. This is valid in JavaScript and we call it a "callback". So a function that is passed to another function as a parameter is a callback function.

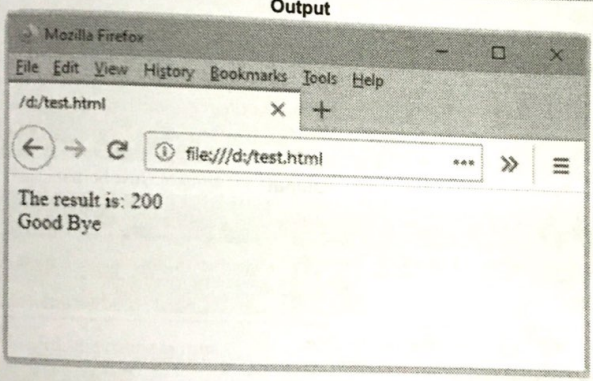
Following JavaScript illustrates the use of callback function

```
<!DOCTYPE html>
<html>
<head>
    <script>
        function mul(a,b,callback)
        {
```

Scanned with CamScanner

```
        ans = a*b
        document.write("The result is: "+ans);
        callback();
    }
    function display(){
        document.write("<br/>Good Bye");
    }
  </script>
</head>
<body>
<script>
  mul(10,20,display);
</script>
</body>
</html>
```

Output



Script Explanation : In above script

(1) We have written a function named **mul**. To this function, a callback function named **display** is passed as a parameter.

(2) After execution of **mul** function, the callback function **display** function gets called, due to **callback()**.

(3) Thus the mechanism of callback function allows to call the functions subsequently.

3.17 Function as Arguments in JavaScript

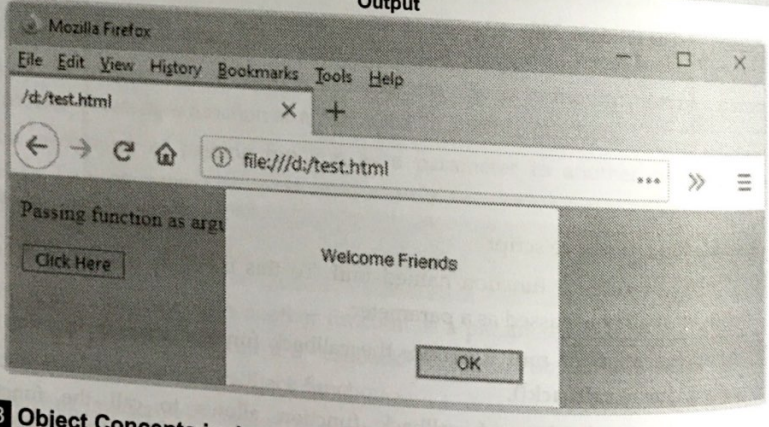
We can pass a function as a argument to another function in the same way as a variable is passed as an argument.

Scanned with CamScanner

Following JavaScript demonstrates how to pass a function as an argument to another function

```
<!DOCTYPE html>
<html>
  <body>
    <p>
      Passing function as arguments.
    </p>
      <button onclick = "OuterFunction(InnerFunction)">
        Click Here
      </button>
    <script>
      function InnerFunction(value){
        return "Welcome Friends";
      }
      function OuterFunction(func){
        alert(func());
      }
    </script>
  </body>
</html>
```

Output



3.18 Object Concepts in JavaScript

In JavaScript object is a collection of properties. These properties are nothing but the members of the classes from Java or C++. For instance - in JavaScript the object Date() is used which happens to the member of the class in Java.

Scanned with CamScanner

3.18.1 Math Objects

- For performing the mathematical computations there are some useful methods available from **math** object.
- For example if we want to find out minimum of two numbers then we can write -

```
document.write(math.min(4.5,7.8));
```

The above statement will result in 4.5. Thus using various useful methods we can perform many mathematical computations.

- Here are some commonly used methods from **math** object.

Method	Meaning
sqrt(num)	This method finds the square root of num.
abs(num)	This method returns absolute value of num.
ceil(num)	This method returns the ceil value of num. For example ceil(10.3) will return 11.
floor(num)	This method returns the floor value of num. For example floor(10.3) will return 10.
log(num)	This method returns the natural logarithmic value of num. For example log(7.9) will return 2.
pow(a,b)	This method will compute the ab. For example pow(2,5) will return 32.
min(a,b)	Returns the minimum value of a and b.
max(a,b)	Returns the maximum value of a and b.
sin(num)	Returns the sine of num.
cos(num)	Returns the cosine of num.
tan(num)	Returns the tangent of num.
exp(num)	Returns the exponential value i.e. enum .

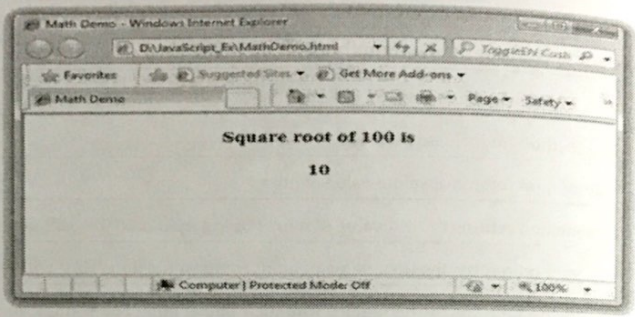
JavaScript Program[MathDemo.html]

```
<!DOCTYPE html>
<html >
<head>
 <title>Math Demo</title>
</head>
<body>
 <center>
  <h3>Square root of 100 is </h3>
  <script type="text/javascript">
```

Scanned with CamScanner

```
  var num=100;
  document.write("<h3>"+Math.sqrt(num)+"</h3>");
  </script>
 </center>
</body>
</html>
```

Output



3.18.2 Number Objects

Various properties of number object are -

Property	Meaning
MAX_VALUE	Largest possible number gets displayed.
MIN_VALUE	Smallest possible number gets displayed.
NaN	When not a number then NaN is displayed.
PI	The value of PI gets displayed.
POSITIVE_INFINITY	The positive infinity gets displayed.
NEGATIVE_INFINITY	The negative infinity gets displayed.

Using **Number.***property_name* we can display the property value. Following JavaScript uses the property of negative infinity.

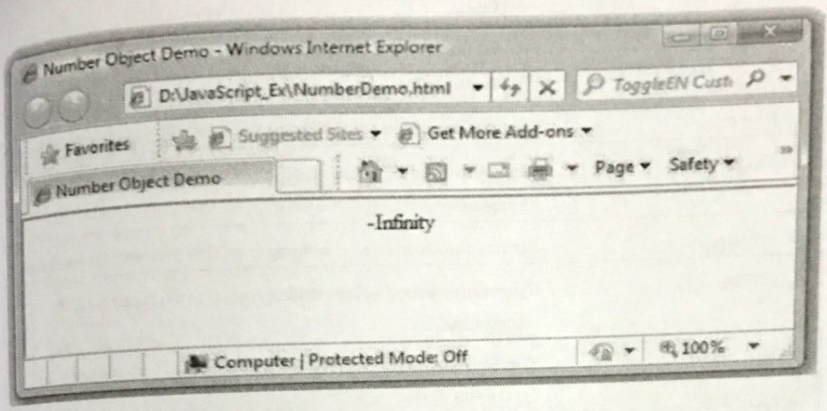
JavaScript[NumberDemo.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Number Object Demo</title>
</head>
<body>
 <center>
  <script type="text/javascript">
```


Scanned with CamScanner

```
document.write(Number.NEGATIVE_INFINITY);
</script>
</center>
</body>
</html>
```

Output



3.18.3 Date Objects

- This object is used for obtaining the date and time.
- This date and time value is based on computer's local time (system's time) or it can be based on GMT(Greenwich Mean Time).
- Nowadays this GMT is also known as UTC i.e. Universal Co-ordinated Time. This is basically a world time standard.
- Following are the commonly used methods of Date object.

Method	Meaning
getTime()	It returns the number of milliseconds. This value is the difference between the current time and the time value from 1st January 1970.
getDate()	Returns the current date based on computers local time.
getUTCDate()	Returns the current date obtained from UTC.
getDay()	Returns the current day. The day number is from 0 to 6 i.e. from Sunday to Saturday.
getUTCDay()	Returns the current day based on UTC. The day number is from 0 to 6 i.e. from Sunday to Saturday.

Scanned with CamScanner

getHours()	Returns the hour value ranging from 0 to 23, based on local time.
getUTCHours()	Returns the hour value ranging from 0 to 23, based on UTC timing zone.
getMilliseconds()	Returns the milliseconds value ranging from 0 to 999, based on local time.
getUTCMilliseconds()	Returns the milliseconds value ranging from 0 to 999, based on UTC timing zone.
getMinutes()	Returns the minute value ranging from 0 to 59, based on local time.
getUTCMinutes()	Returns the minute value ranging from 0 to 59, based on UTC timing zone.
getSeconds()	Returns the second value ranging from 0 to 59, based on local time.
getUTCSeconds()	Returns the second value ranging from 0 to 59, based on UTC timing zone.
setDate(value)	This function helps to set the desired date using local timing or UTC timing zone.
setHour(hr,minute,second,ms)	This function helps to set the desired time using local or UTC timing zone. The parameters that can be passed to this function are hour,minute,seconds and milliseconds. Only hour parameter is compulsory and rest all are the optional parameters.

In the following web document we are making use of **Date()** object and some useful methods of it.

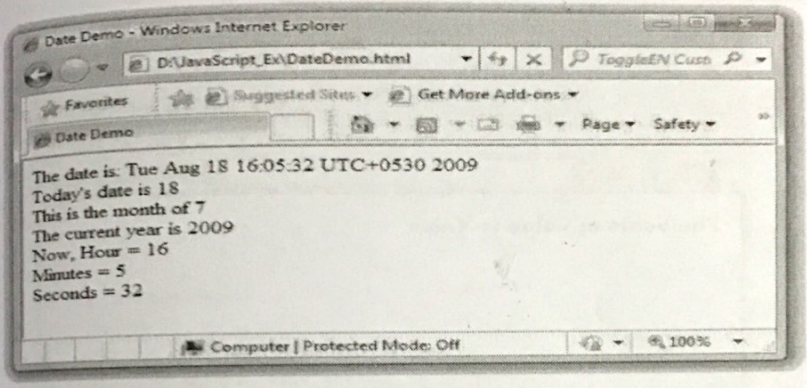
JavaScript[DateDemo.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Date Demo</title>
</head>
<body>
  <script type="text/javascript">
  var my_date=new Date();
  document.write("The date is: "+my_date.toString()+"<br>");
  document.write("Today's date is "+my_date.getDate()+"<br>");
  document.write("This is the month of "+my_date.getMonth()+"<br>");
  document.write("The current year is "+my_date.getFullYear()+"<br>");
  document.write("Now, Hour = "+my_date.getHours()+"<br>");
```


Scanned with CamScanner

```
document.write("Minutes = "+my_date.getMinutes()+"<br>");
document.write("Seconds = "+my_date.getSeconds()+"<br>");
</script>
</body>
</html>
```

Output



Script Explanation :

1. In the above script we have created an instance *my_date* of the object **Date().**
2. Then using **my_date.toString()** method we can display the current date along with the time.
3. Then using the functions like getDate(), getMonth(), getFullYear() we are displaying the date, month and year separately.
4. Similarly using the functions getHours(), getMinutes() and getSeconds() we can display the current hour, minute and seconds separately.

3.18.4 Boolean Objects

- This object is the simplest kind of object which is used especially when we want to represent **true** and **false** values.
- Here is a simple javascript in which the Boolean type variable is used -

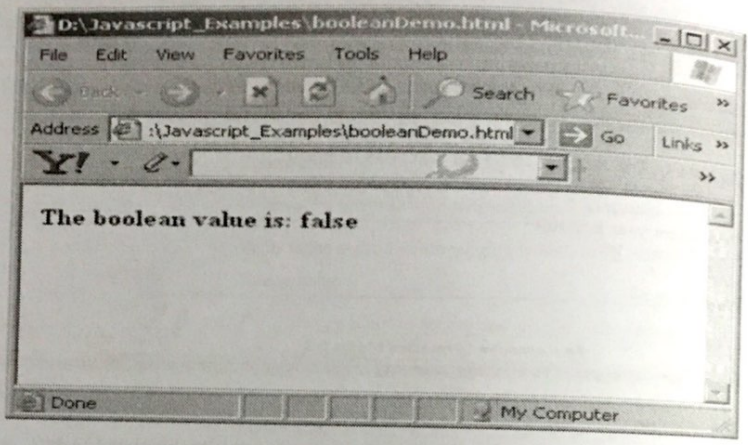
Javascript Program

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


Scanned with CamScanner

```
var temp=new Boolean(false);
document.write("<b>"+"The boolean value is:  ");
document.write(temp.toString());
</script>
</body>
</html>
```

Output



3.18.5 String Objects

- String is a collection of characters.
- In JavaScript using **string** object many useful string related functionalities can be exposed off.
- Some commonly used methods of string object are concatenating two strings, converting the string to upper case or lower case, finding the substring of a given string and so on.
- Here is a listing of some methods of string.

Method	Meaning
concat(str)	This method concatenates the two strings. For example s1.concat(s2) will result in concatenation of string s1 with s2.
charAt(index_val)	This method will return the character specified by value index_val.
substring(begin,end)	This method returns the substring specified by begin and end character.
toLowerCase()	This function is used to convert all the uppercase letters to lower case.

Scanned with CamScanner

toUpperCase()	This function is used to convert all the lowercase letters to upper case.
valueOf()	This method returns the value of the string.

There is one important property of string object and that is **length**. For example

```
var my_str="Hello";
var len;
len=my_str.length;
```

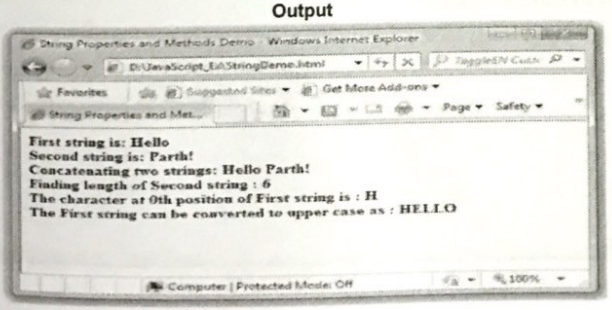
Length of the string "Hello" will be stored in the variable len

Here is sample program in which some methods of string object are used.

JavaScript[StringDemo.html]

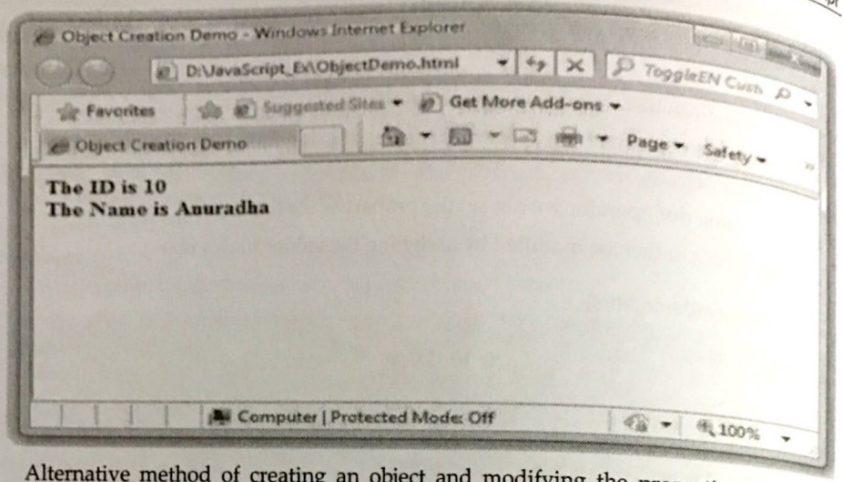
```
<!DOCTYPE html>
<html>
<head>
  <title>String Properties and Methods Demo</title>
</head>
<body>
<strong>
  <script type="text/javascript">
   var s1="Hello ";
   var s2="Parth!";
   document.write("First string is: "+s1+"<br>");
   document.write("Second string is: "+s2+"<br>");
   document.write("Concatenating two strings: "+s1.concat(s2)+"<br>");
   document.write("Finding length of Second string : "+s2.length+"<br>");
   document.write("The character at 0th position of First string is : "+s1.charAt(0)+"<br>");
   document.write("The First string can be converted to upper case as : "+s1.toUpperCase());
  </script>
 </strong>
</body>
</html>
```

Output



Scanned with CamScanner

Output



Alternative method of creating an object and modifying the properties is as given below -

```
var Myobj={name:"Anuradha",id:10};
```

Then using **document.write** statement we can display the property values as follows -

```
document.write("The ID is "+Myobj.id+"<br/>");
document.write("The Name is "+Myobj.name);
```

The property of created object can be deleted using an expression **delete**. Normally the property of an object is deleted in order to free the allocated memory so that this memory can be reused by other process.

3.19 JSON

- JSON stands for JavaScript Object Notation.
- Using JSON we can store and retrieve data. This text based open standard format.
- It is extended from JavaScript language.

Features of JSON

1. It is **text based, lightweight data interchange format.**
2. It is **language independent.**
3. It is **easy** to **read** and **write.**
4. It is **easy** for **machines** to **parse** and generate.
5. It uses the conventions that are **familiar** to the languages like C, C++, Java, JavaScript, Perl, Python and so on.

Scanned with CamScanner

3.19.1 Syntax

JSON is built on **two structures :**

1. A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
2. An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

JSON Object

- JSON object holds the Key value pair.
- Each key is represented as string and value can be of any datatype.
- The key and value are separated by colon.
- **Syntax**

```
{ string : value, .....}
```

- **For example**

```
"Age":38
```

- Each key value pair is separated by comma.
- The object is written within the { }curly brackets.

1) JSON object containing value of different data types

```
{
  "student":
  {
    "name":    AAA",      ← String value
    "roll_no":  10,       ← Numeric value
    "Indian":   true      ← Boolean value
  }
}
```

2) JSON Nested Object

```
{
  "student":
  {
    "name":    "AAA",
    "roll_no":  10,
    "address":
      {
      "Street": "Shivaji Nagar",
```


Scanned with CamScanner

```
        "City": "Pune",
        "Pincode": 411005
        }
    }
}
```

3) Creation of JSON Object with JavaScript

In JavaScript we can write the JSON object and store it in some variable. For example

```
var stud = { "name":"AAA", "roll_no":10, "city":"Pune" };
```

Here the object named stud is created.

Example

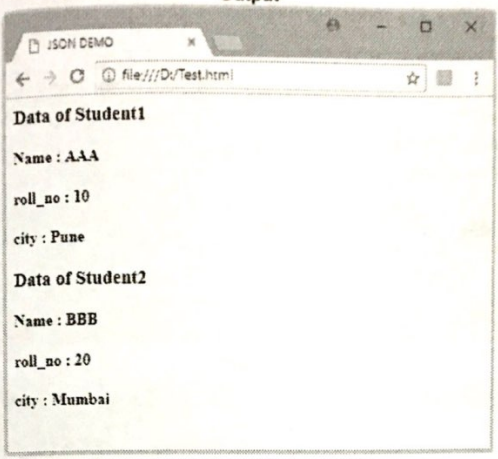
Let us understand how to create an JSON object and obtain its value, and get it displayed on the web browser using JavaScript. The code is as follows -

Test.html

```
<html>
  <head>
    <title>JSON DEMO</title>
    <script language = "javascript" >
      var stud1 = { "name":"AAA", "roll_no":10, "city":"Pune"};
      document.write("<h3>Data of Student1</h3>");
      document.write("<h4>Name : " + stud1.name+"</h4>");
      document.write("<h4>roll_no : " + stud1.roll_no+"</h4>");
      document.write("<h4>city : " + stud1.city+"</h4>");

      var stud2 = { "name":"BBB", "roll_no":20, "city":"Mumbai"}
      document.write("<h3>Data of Student2</h3>");
      document.write("<h4>Name : " + stud2.name+"</h4>");
      document.write("<h4>roll_no : " + stud2.roll_no+"</h4>");
      document.write("<h4>city : " + stud2.city+"</h4>");
    </script>
  </head>
  <body>
  </body>
</html>
```

Scanned with CamScanner

Output



JSON with Arrays

JSON array represents the collection of values. It is denoted within [ ]. The elements of array are separated by comma.

Syntax

```
[ value, .....]
```

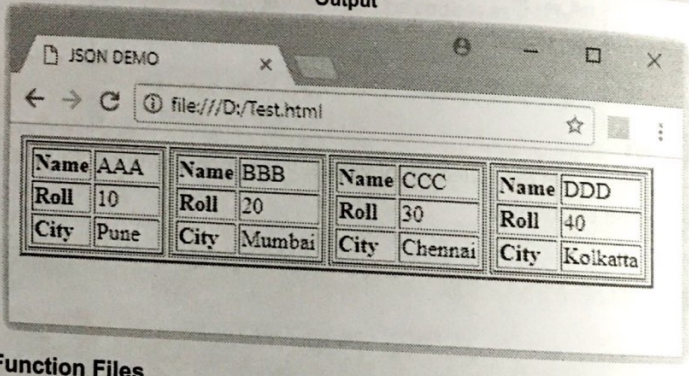
Example

```
<html>
  <head>
   <title>JSON DEMO</title>
   <script language = "javascript" >
    var obj = {"Student":[
        { "name":"AAA", "roll_no":10, "city":"Pune"},
        { "name":"BBB", "roll_no":20, "city":"Mumbai"},
        { "name":"CCC", "roll_no":30, "city":"Chennai"},
        { "name":"DDD", "roll_no":40, "city":"Kolkatta"}
        ]};
var i = 0
      document.writeln("<table border = '2'><tr>");
   for(i = 0;i<obj.Student.length;i++)
   {
        document.writeln("<td>");
        document.writeln("<table border = '1' width = 100 >");
        document.writeln("<tr><td><b>Name</b></td><td width = 50>"+
```


Scanned with CamScanner

```
        obj.Student[i].name+"</td></tr>");
        document.writeln("<tr><td><b>Roll</b></td><td width = 50>"
obj.Student[i].roll_no
        +"</td></tr>");
      document.writeln("<tr><td><b>City</b></td><td width = 50>" + obj.Student[i].city
      +"</td></tr>");
      document.writeln("</table>");
      document.writeln("</td>");
    }
  </script>
 </head>
 <body>
 </body>
</html>
```

Output



3.19.2 Function Files

Using JSON, it is possible to define a function in a separate external JS file and we can access this functionality from HTML document.

Following example illustrates this idea

Example Code

Step 1 : We can write a JavaScript file containing an array of Student's information. This file is saved with .js extension. The code for it is as follows -

myFile.js

```
Student([
      { "name":"AAA", "roll_no":10, "city":"Pune"},
      { "name":"BBB", "roll_no":20, "city":"Mumbai"},
      { "name":"CCC", "roll_no":30, "city":"Chennai"},
      { "name":"DDD", "roll_no":40, "city":"Kolkatta"}
])
```

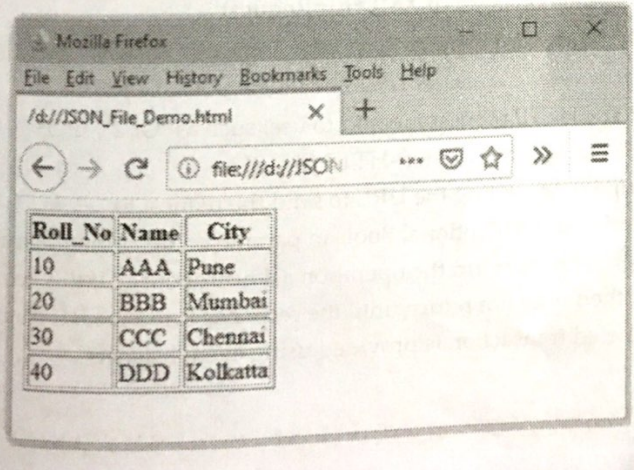

Scanned with CamScanner

Step 2 : Now we will invoke this file in our JSON script to access the elements of an array. The code is as follows -

JSON_File_Demo.html

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

<div id="myID"></div>
<script>
function Student(arr) {
 var out = "";
 var i;
 out+='<table border=1>'
 out+='<tr><th>Roll_No</th><th>Name</th><th>City</th></tr>'
 for(i = 0; i<arr.length; i++) {
 out += '<tr><td>'+arr[i].roll_no + '</td><td>' +arr[i].name + '</td><td>' +arr[i].city +
'</td></tr>';
 }
 out+='</table>'
 document.getElementById("myID").innerHTML = out;
}
</script>
<script src="myFile.js"></script> // Invoking external file

</body>
</html>
```

Output



Scanned with CamScanner

Script Explanation :

In above script,

1. We have written the Student function in a file named **MyFile.js**
2. This function defines an array of three fields Roll_No, Name and City. There are four elements in that array
3. In our HTML file, the JSON code refers to the external JS file as shown below

```
<script src="myFile.js"></script>
```

4. Then the elements of array are accessed and displayed on the browser as follows

```
for(i = 0; i<arr.length; i++) {
    out += '<tr><td>'+arr[i].roll_no +
    '</td><td>' +arr[i].name +
    '</td><td>' +arr[i].city + '</td></tr>';
}
```

3.19.3 HTTP Request

- JSON is most commonly used in asynchronous HTTP requests. This is where an application pulls data from another application via an HTTP request on the web.
- **XMLHttpRequest** is an API that provides scripted client functionality for transferring data between a client and a server.
- It allows you to get data from an external URL without having to refresh the page

XMLHttpRequest includes a number of methods and attributes.

- We use two functions of XMLHttpRequest - **Open()** and **send()**.
- The **open()** to initialize the request, and **send()** to send the request.
- The Syntax of method **open of XMLHttpRequest** is

```
XMLHttpRequest.open(method, url[, async[, user[, password]]])
```

where

- **Method :** The HTTP request method to use, such as "GET", "POST", "PUT", "DELETE", etc. Ignored for non-HTTP(S) URLs.
- **url :** A string representing the URL to send the request to.
- **async (Optional) :** An optional Boolean parameter, defaulting to true, indicating whether or not to perform the operation asynchronously. If this value is false, the send() method does not return until the response is received. If true, notification of a completed transaction is provided using event listeners.

Scanned with CamScanner

- Following code illustrates this idea

Step 1 : Create a text file as

myfile.txt

```
[
    { "name":"AAA", "roll_no":10, "city":"Pune"},
    { "name":"BBB", "roll_no":20, "city":"Mumbai"},
    { "name":"CCC", "roll_no":30, "city":"Chennai"},
    { "name":"DDD", "roll_no":40, "city":"Kolkatta"}
]
```

Step 2 : The JSON code to invoke above text file using Http Request is as follows -

JSON_HttpReq_Demo.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
</head>
<body>

<div id="myID"></div>
<script>

var xmlhttp = new XMLHttpRequest();
var url = "file:myfile.txt";

xmlhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        var stud_arr = JSON.parse(this.responseText);
        Student(stud_arr);
    }
};

xmlhttp.open("GET", url, true);
xmlhttp.send();

function Student(arr) {
```


Scanned with CamScanner

```
var out = "";
var i;
out+='<table border=1>'
out+='<tr><th>Roll_No</th><th>Name</th><th>City</th></tr>'
for(i = 0; i<arr.length; i++) {
out += '<tr><td>'+arr[i].roll_no + '</td><td>' +arr[i].name + '</td><td>' +arr[i].city +
'</td></tr>';
}
out+='</table>'
document.getElementById("myID").innerHTML = out;
}
</script>
</body>
</html>
```

The output as obtained in previous section is displayed on the browser.

❑❑❑

Scanned with CamScanner

5.1 Need of Session Management

- Session simply means particular interval of time. Session tracking is a technique in which particular state of the user is identified and maintained.
- As HTTP is a stateless protocol, it is difficult to track the state of user communication with web. Hence there is a need to track the user state by a special technique. This is called **session management**.
- Session management is needed to identify particular user.

5.2 Various Techniques for State and Session Management

GTU : Summer-12, Marks 10

- A web application consists of **series of disconnected HTTP requests** to web server where each request for server page is basically a request to run a **separate program.**
- There are many occasions when we want the web server to connect the requests together. For instance – In an application like online purchasing system, the user adds an item to the cart, makes online payment and then checks out. In this situation the web server should recognize that the request is from the same individual and these are altogether the subsequent requests to perform the single task of **online purchasing.**
- These requests are made by using HTTP (Hypertext Transfer Protocol) protocol. The **drawback** of this protocol is that it **can not recognize** that the requests that are coming to the web server are from single source or from two different sources. Hence HTTP protocol is also called as **stateless protocol.** It cannot remember from which user the request has come from.
- **To overcome this drawback** of HTTP protocol there are variety of solutions. Out of which the most commonly used techniques of **passing the information** from one request to another are –
  - Using Query String
  - Using URL path
  - Cookies
  - Session Handling

Let us discuss these solutions with illustrative examples.

5.2.1 Hidden Fields

The hidden form field is a technique in which hidden or invisible textfield is used for maintaining the state of the user.

In such case, we store the information in the hidden field and get it from another web page.

Scanned with CamScanner

A hidden field let web developers include data that cannot be seen or modified by users when a form is submitted.

The **<input type="hidden">** defines a hidden input field.

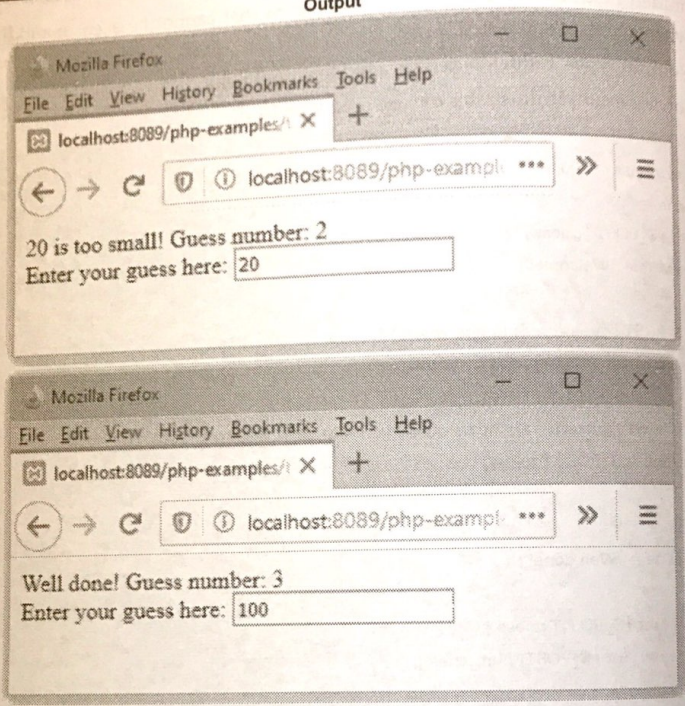
Following example illustrates the use of hidden field.

```
<?php
$num_to_guess = 100;
$message = "";
if (! isset ( $_POST ['guess'] )) {
   $message = "Welcome!";
}
else if ($_POST ['guess'] > $num_to_guess) {
   $message = $_POST ['guess'] . " is too large!";
}
else if ($_POST ['guess'] < $num_to_guess) {
   $message = $_POST ['guess'] . " is too small!";
}
else {
   $message = "Well done!";
}
$guess = ( int ) $_POST["guess"];
$num_tries = ( int ) $_POST["num_tries"];
$num_tries++;
?>
<html>
<body>
<?php print $message?>
Guess number: <?php print $num_tries?><br />
<form method="post"
action="<?php print $_SERVER ['PHP_SELF']?>">
<input type="hidden" name="num_tries"
value="<?php print $num_tries?>" /> Enter your guess here:
<input type="text" name="guess" value="<?php print $guess?>" />
</form>
</body>
</html>
```

Hidden Field

Scanned with CamScanner

Output



5.2.2 Query String

- There are two methods namely – GET and POST for passing the query strings from browser to server.
- Method GET is used to send the query which is less secure. Method POST is used to send more secured data.

Difference between GET and POST Methods

Sr. No.	GET Request	POST Request
1.	**Parameters** remain in **browser history** because they are part of the URL.	**Parameters** are **not saved in browser history.**

Scanned with CamScanner

2.	GET is **less secure** compared to POST because data sent is part of the URL. So it is saved in browser history and server logs in plaintext.	POST is a little **safer than GET** because the parameters are not stored in browser history or in web server logs.
3.	This request can be cached.	This requests are never cached.
4.	This request can be **bookmarked.**	This request **can not be bookmarked.**
5.	GET method **should not be** used when sending passwords or other **sensitive information.**	POST method **used when sending passwords** or other sensitive information.
6.	Only **limited amount of information is sent** using GET request.	**Large amount of information** is sent using POST request.
7.	**It is more efficient.**	**It is less efficient.**

Working of GET and POST

- If we have a input form on which there are some text boxes placed. The user fills up the information within the text boxes and clicks the submit button. The information within the text boxes will then be transmitted to the server via **query string**. This scenario can be represented by following figure –

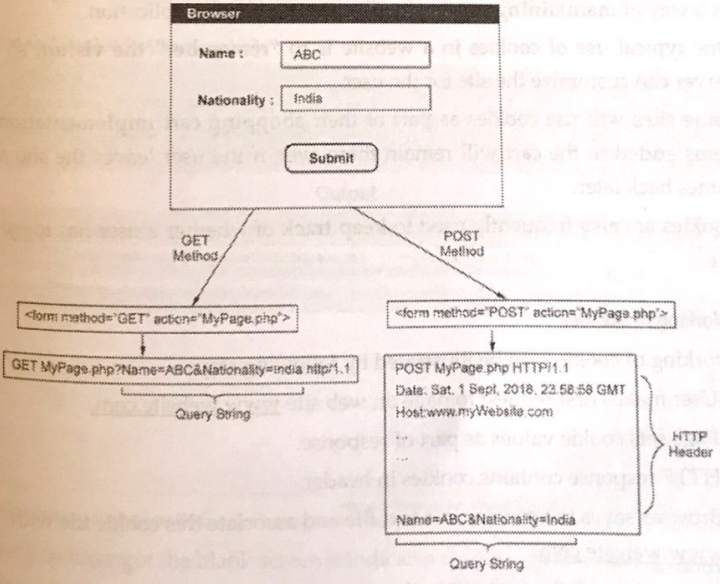


Fig. 5.2.1 Information processing by GET and POST methods

Scanned with CamScanner

Using Query string the information is transmitted to the server.

Review Question

1. Explain the method of passing information via query strings.

5.2.3 Cookies

- Cookie is a small file that server embeds in the user's machine. This is another method of passing user information to the server.
- 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.

Uses of Cookies

- While cookies can be used for any state-related purpose, they are principally used as a way of **maintaining continuity** over time in a web application.
- One typical use of cookies in a website is to **"remember" the visitor**, so that the server can customize the site for the user.
- Some sites will use cookies as part of their **shopping cart implementation** so that items added to the cart will remain there even if the user leaves the site and then comes back later.
- Cookies are also frequently used to **keep track** of whether a user has **logged into a site**.

5.2.3.1 Working of Cookies

The working of cookies can be illustrated by following steps –

Step 1 : User makes first request to page on web site www.website.com.

Step 2 : Page sets cookie values as part of response.

Step 3 : HTTP response contains cookies in header.

Step 4 : Browser saves the cookies in a text file and associate this cookie file with www.website.com.

Step 5 : User makes another request to the page on the site www.website.com.

Scanned with CamScanner

Step 6 : Browser reads cookie values from text file for each subsequent request for www.website.com.

Step 7 : Cookie values travel in every subsequent HTTP request for that domain.

Step 8 : Server for www.website.com retrieves these cookie values from request header and uses them to customize the response.

- There are two types of cookies : **session cookies** and **persistent cookies.**
- A **session cookie** has no expiry stated and thus will be deleted at the end of the user browsing session.
- **Persistent cookies** have an expiry date specified; they will persist in the browser's cookie file until the expiry date occurs, after which they are deleted.

5.2.3.2 Using 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 -

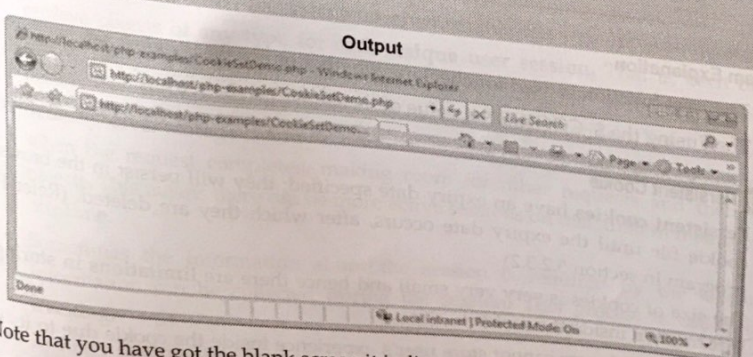
```
setcookie(name,value,expire_period,path,domain)
```

- 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.

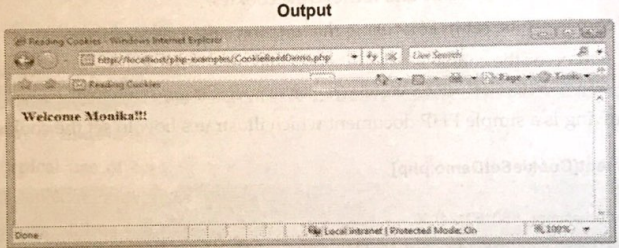
Scanned with CamScanner

Now you can retrieve the cookie and read the value to ensure whether or not 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.

5.2.3.3 Persistent Cookie

- **Persistent cookies** have an expiry date specified; they will persist in the browser's cookie file until the expiry date occurs, after which they are deleted. (Refer PHP program in section 5.2.3.2)
- The size of cookies is very very small and hence there are **limitations in storing of information** inside the cookie.
- The web developers cannot store user's experience inside the cookie due to its **size and reliability constraints.**

Scanned with CamScanner

- Many sites provide a **Remember Me** checkbox on the login page in order to store user name. This facility makes use of **persistent cookie**.
- Instead, the login cookie would contain a **random token**. This random token would be stored along with the username in the site's back-end database. Every time the user logs in, a new token would be generated and stored in the database and cookie.
- Another secondary **use of cookies** is to **store user's preferences**. For instance some user's prefer to use particular site language, site color and so on. In this situation using persistent cookies the web developers store the user's preference and give contented experience to the user.
- The persistent cookie can also **save user's browser behavior** on a site. This information can be used by the site administrator as an analytic tool to help **understand** how users **navigate through the site.**

Review Questions

1. *Explain the concept of cookies with its working.*
2. *Write a PHP script to demonstrate creation and reading of cookies.*
3. *What is persistent cookies ?*

5.2.4 Session

- When you open some application, use it for some time and then close it. This entire scenario is named as **session.**
- **Session state** is a server-based state mechanism that lets web applications store and retrieve objects of any type for each **unique user session.** That is, each browser session has its own session state stored as a serialized file on the server, which is deserialized and loaded into memory as needed for each request. Refer Fig. 5.2.2.
- Because server storage is a finite resource, objects loaded into memory are released when the request completes, making room for other requests and their session objects. This means there can be more active sessions on disk than in memory at any one time.
- 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.**
- In PHP, session state is available to the developer using the $_SESSION variable. The unique ID for the sessions can be stored in superglobal array $_SESSION.

Scanned with CamScanner

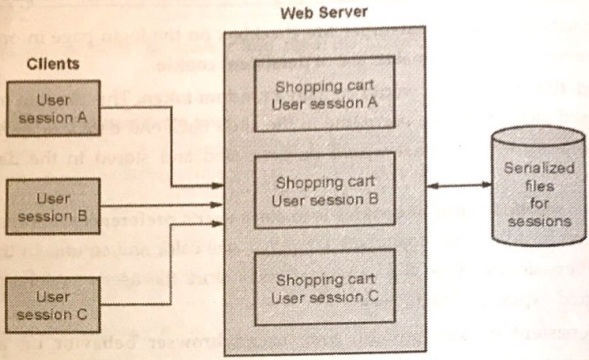


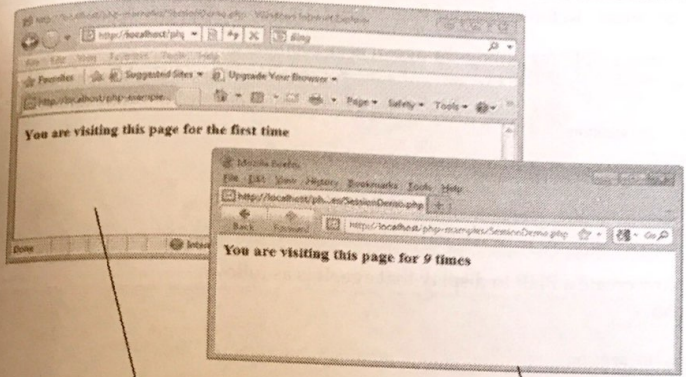
Fig. 5.2.2 Session state

- 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>";
}
?>
```

Scanned with CamScanner



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.

Difference between Cookie and Session variable

Sr. No.	Cookie	Session
1.	The data stored in cookie will survive for desired period.	The data stored in session gets destroyed when the session ends.
2.	Cookie is stored on local machine.	Session variable will not be stored in local machine.
3.	The cookies are normally used at client side.	The sessions normally used at server side.

Example 5.2.1 *What is session ? How do PHP manage the sessions ? Give an example to store and retrieve, your age as session variable.* **GTU : Summer-12, Marks 10**

Scanned with CamScanner

Solution : Sessions - Refer section 5.2.5.

Step 1 : Create a PHP for setting the age. It is as follows -

SetAge.php

```
<?php
// this starts the session
session_start();
 // this sets variables in the session
$_SESSION['Age']='35';
print "Done";
?>
```

Step 2 : Now create a PHP to display that age. It is as follows -

ShowAge.php

```
<?php
// this starts the session
session_start();
// echo variable from the session, we set this on our other page
echo "My Age is: ".$_SESSION['Age'];
?>
```

❑❑❑

Scanned with CamScanner

For example

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("mydb",$conn); //select the database
$query=" DELETE FROM my_table WHERE id=1";
mysql_query($query,$conn);//Execution of Query
mysql_close($conn); //closing the database
?>
```

Similarly we can delete a database using the query DROP.

For example –

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","mypassword");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("mydb",$conn); //select the database
$query=" DROP DATABASE mydb";
mysql_query($query,$conn);//Execution of Query
mysql_close($conn); //closing the database
?>
```

6.4 Object Oriented Database Access using PHP

GTU : Summer 17, Marks 7

Example 6.4.1 *Write a PHP script to create a new database table with 4 fields of your choice and perform following database operations.*
i) Insert ii) Update iii) Delete

Solution : We will create a table in the database **test**. The name of the table is **mytable**. Then we will insert the record into the table using the INSERT command,update particular field of the record using the command UPDATE and Delete the record using the command DELETE.

The PHP script is as follows -

PHP Document[DBDemo.php]

```
<?php
```

Scanned with CamScanner

```
// Make a MySQL Connection
mysql_connect("localhost", "root", "mypassword") or die(mysql_error());
mysql_select_db("test") or die(mysql_error());
echo "Connected to database!";
mysql_query("CREATE TABLE mytable(id INT NOT NULL AUTO_INCREMENT, PRIMARY KEY(id),
name VARCHAR(30),
 phone INT,emailId VARCHAR(30))")
 or die(mysql_error());
print "<br/>";
echo "Table Created!";

// Insert a row of information into the table "example"
mysql_query("INSERT INTO mytable
(name, phone,emailId) VALUES('Priyanka', '11111','abc123@gmail.com' ) ")
or die(mysql_error());

mysql_query("INSERT INTO mytable
(name, phone,emailId) VALUES('Kumar', '22222','pqr11@yahoo.com' ) ")
or die(mysql_error());

mysql_query("INSERT INTO mytable
(name, phone,emailId) VALUES('Archana', '33333','xyz@rediffmail.com' ) ")
or die(mysql_error());
print "<br/>";
echo "Data Inserted!";
$result =mysql_query("SELECT * FROM mytable")
or die(mysql_error());
print "<br/>";
print "<b>User Database</b>";
echo "<table border='1'>";
echo "<tr><th>ID</th> <th>Name</th> <th>Phone</th><th>Email-ID</th> </tr>";
while($row = mysql_fetch_array( $result ))
    {
    // Print out the contents of each row into a table
    echo "<tr><td>";
    echo $row['id'];
    echo "</td><td>";
    echo $row['name'];
    echo "</td><td>";
    echo $row['phone'];
    echo "</td><td>";
    echo $row['emailId'];
    echo "</td></tr>";
}
```

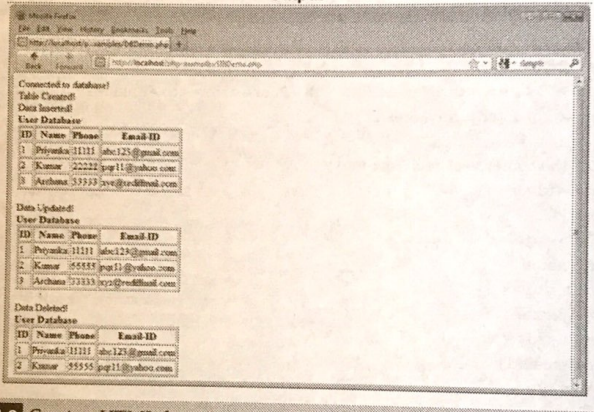

Scanned with CamScanner

```
echo "</table>";
$result = mysql_query("UPDATE mytable SET phone='55555' WHERE phone='22222'")
or die(mysql_error());

print "<br/>";
echo "Data Updated!";
$result =mysql_query("SELECT * FROM mytable")
or die(mysql_error());
print "<br/>";
print "<b>User Database</b>";
echo "<table border='1'>";
echo "<tr><th>ID</th> <th>Name</th> <th>Phone</th><th>Email-ID</th> </tr>";
while($row = mysql_fetch_array( $result ))
    {
    // Print out the contents of each row into a table
    echo "<tr><td>";
    echo $row['id'];
    echo "</td><td>";
    echo $row['name'];
    echo "</td><td>";
    echo $row['phone'];
    echo "</td><td>";
    echo $row['emailId'];
    echo "</td></tr>";
}
echo "</table>";
$result = mysql_query("DELETE FROM mytable WHERE phone='33333'")
or die(mysql_error());
print "<br/>";
echo "Data Deleted!";
$result =mysql_query("SELECT * FROM mytable")
or die(mysql_error());
print "<br/>";
print "<b>User Database</b>";
echo "<table border='1'>";
echo "<tr><th>ID</th> <th>Name</th> <th>Phone</th><th>Email-ID</th> </tr>";
while($row = mysql_fetch_array( $result ))
    {
    // Print out the contents of each row into a table
    echo "<tr><td>";
    echo $row['id'];
    echo "</td><td>";
    echo $row['name'];
    echo "</td><td>";
```


Scanned with CamScanner

```
    echo $row['phone'];
    echo "</td><td>";
    echo $row['emailId'];
    echo "</td></tr>";
}
echo "</table>";
?>
```

Output



Example 6.4.2 *Create a HTML form "result.html" with a text box and a submit button to accept registration number of the student. Write a "result.php" code to check the status of the result from the table to display whether the student has "PASS" or "FAIL" status. Assume that the MYSQL database "my_db" has the table "result_table" with two columns REG_NO and STATUS.*

Also write a PHP program to delete a record from result_table.

Solution :

Step 1 : Create a database named my_db. Create a table result_table for this database and insert the values in this table. The table is created as follows –

Scanned with CamScanner

REG_NO	STATUS
101	PASS
102	FAIL
103	PASS
104	FAIL
105	PASS

Step 2 : Create an HTML form to accept the registration number, the HTML document is as follows –

result.html

```
<!DOCTYPE html>
<html>
  <head>
    <title> STUDENT RESULT </title>
  </head>
  <body>
    <form name="myform" method="post" action="http://localhost/php-examples/result.php">
      <input type="text" name="reg_no"/>
      <input type="submit" value="Submit"/>
    </form>
  </body>
</html>
```

Step 3 : Create a PHP script to accept the registration number. This php script will connect to MYSQL database and the status(PASS or FAIL) of the corresponding registration number will be displayed.

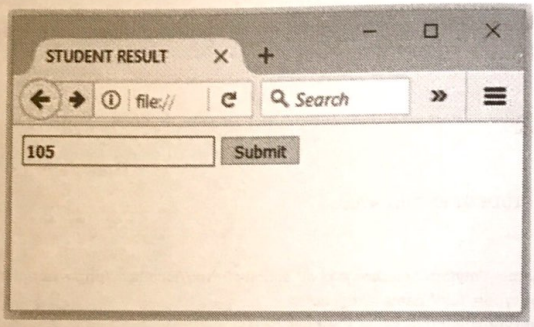
result.php

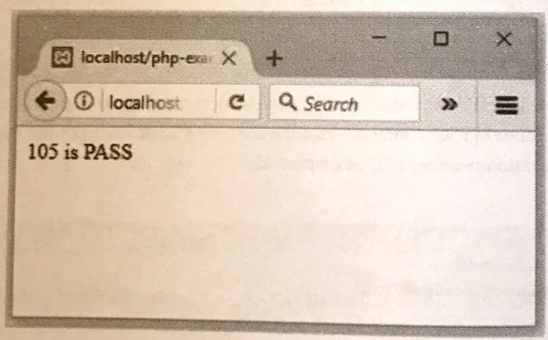
```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("my_db",$conn); //select the database
//Execution of Query for displaying the data
$reg_no = intval($_POST['reg_no']);

$result=mysql_query("SELECT REG_NO,STATUS FROM result_table where REG_NO=$reg_no ");
while($row = mysql_fetch_array($result))
```


Scanned with CamScanner

```
 {
  echo $row['REG_NO'] . " is " . $row['STATUS'];
  echo "<br />";
 }
mysql_close($conn); //closing the database
?>
```

Step 4 : Load the HTML form created in Step 2 and click the submit button by entering some registration number.





Step 4 : Deletion of record when user submits the registration number.

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","");
if(!$conn)
{
die('error in connection'.mysql_error());
}
```


Scanned with CamScanner

```
mysql_select_db("my_db",$conn); //select the database
//Execution of Query for displaying the data
$reg_no = intval($_POST['reg_no']);

$result=mysql_query("DELETE FROM result_table where REG_NO=$reg_no");
while($row = mysql_fetch_array($result))
{
echo $row['REG_NO'] . " is " . $row['STATUS'];          //Each record will be displayed
                                                        //line by line
echo "<br />";
}

mysql_close($conn); //closing the database
?>
```

Example 6.4.3 *Write a user defined function 'CalculateInterest' using PHP to find the simple interest to be paid for a loan amount. Read the loan amount, the number of years and rate of interest from a database table called LOANDETAILS having three fields AMT, YEARS, and RATE, and Calculate the interest using the user defined function.*

Solution :

Step 1 : Create a database table named LOANDETAILS having three fields AMT, YEARS and RATE. Insert the values in it. The sample table will be as follows –

AMT	YEARS	RATE
10000	5	6.9

Step 2 : The PHP code for calculating the simple interest the above values in a function will be as follows –

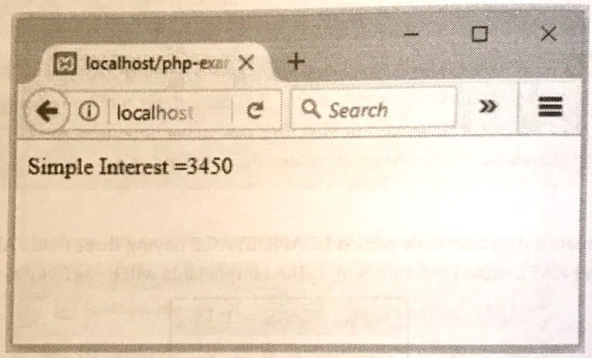
Interest.php

```
<?php
function CalculateInterest()
{
   // Make a MySQL Connection
   $conn=mysql_connect("localhost","root","");
   if(!$conn)
   {
      die('error in connection'.mysql_error());
   }
   mysql_select_db("my_db",$conn); //select the database
   //Execution of Query for displaying the data
```


Scanned with CamScanner

```
    $result=mysql_query("SELECT * FROM LOANDETAILS");
    $row = mysql_fetch_array($result);
    $amount= $row['AMT'];
    $rate= $row['RATE'];
    $years= $row['YEARS'];
     $interest=($amount * $rate *$years)/100;
    mysql_close($conn); //closing the database
    return $interest;
}
print "Simple Interest =".CalculateInterest();
?>
```

Output



Example 6.4.4 *Write HTML and PHP program to read and store student information such as enrollment no, name, semester and percentage from database table. Output the data to a webpage in tabular format.* **GTU : Summer 17, Marks 7**

Solution:

Step 1 : Create an HTML form as follows to input the student info.

Input_student_info.html

```
<html>
<head>
    <title> Inserting Data into Database Table</title>
</head>
<body>
    <form method = "post" action = "http://localhost/php-examples/studentInfo.php ">
    <label>Enrollment No</label>
    <input type="text" name="enrollment_no" />
    <br />
```


Scanned with CamScanner

```
    <label>Name</label>
    <input type="text" name="name" />
    <br />
    <label>Semester</label>
    <input type="text" name="semester" />
    <br />
    <label>Percentage</label>
    <input type="text" name="percentage" />
    <br />
    <input type="submit" value="Submit">
</form>
</body>
</html>
```

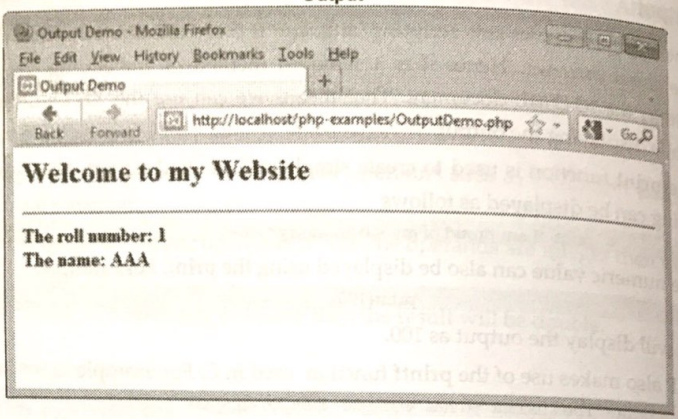
Step 2 : Now we will create a PHP script in which the data is received from the above HTML form and inserted in the database table. This database is then displayed in tabular form on the web page. The PHP code for this is as follows –

studentInfo.php

```
<?php
// Make a MySQL Connection
$conn=mysql_connect("localhost","root","");
if(!$conn)
{
die('error in connection'.mysql_error());
}
mysql_select_db("StudentDB",$conn); //select the database
$query="INSERT INTO student_table (enrollment_no,name,semester,percentage)
VALUES('$_POST[enrollment_no]','$_POST[name]' ,'$_POST[semester]','$_POST[percentage]')";
mysql_query($query,$conn);//Execution of Query
$result=mysql_query("SELECT * FROM student_table");
echo "<table border='1'>";
while($row = mysql_fetch_array($result))
{
 echo "<tr>";
 echo    "<td>".$row['enrollment_no']    .    "</td><td>"    .    $row['name']    ."</td><td>"    .
$row['semester']."</td><td>" . $row['percentage'];
 //Each record will be displayed
  echo "</td></tr>";
}
echo "</table>";
mysql_close($conn); //closing the database
?>
```

Scanned with CamScanner

Output



4.5 Decision and Looping with Examples

GTU : Winter-11,12,14, Summer-13,14,16, Marks 7

- 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.

4.5.1 Selection Statements

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

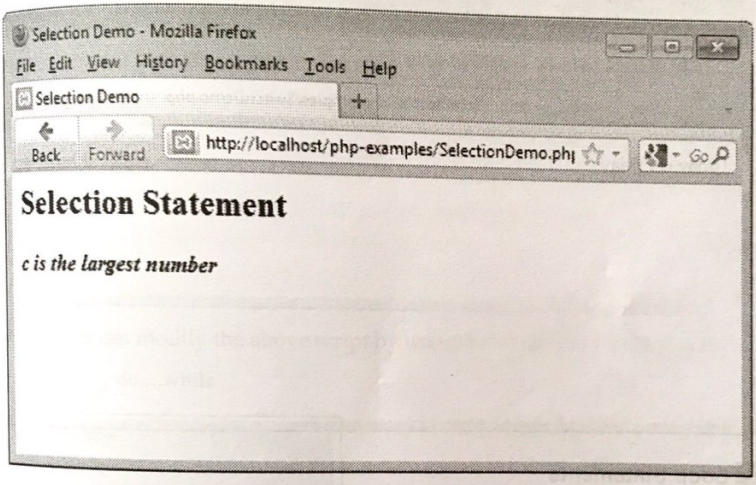
PHP Document[selection.php]

```
<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>";
```

Scanned with CamScanner

```
    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

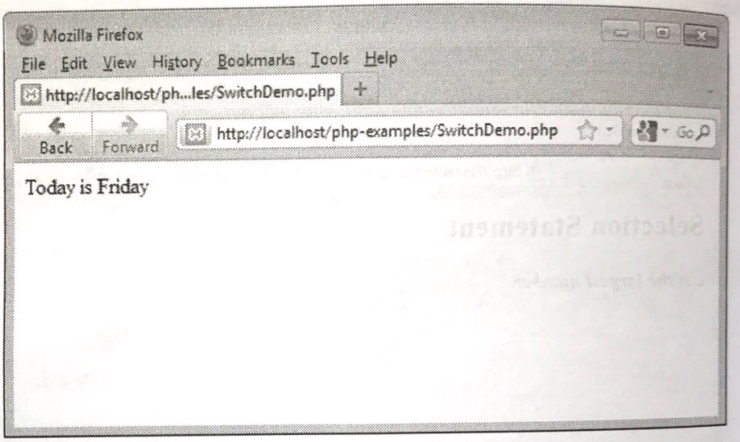
PHP Document[SwitchDemo.php]

```
<?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;
```

Scanned with CamScanner

```
        case "Saturday":print "Today is Saturday";
                                                    break;
        case "Sunday":print "Today is Sunday ";
                                                    break;
    default: print "Invalid input";
}
?>
```

Output



4.5.2 Loop Statements

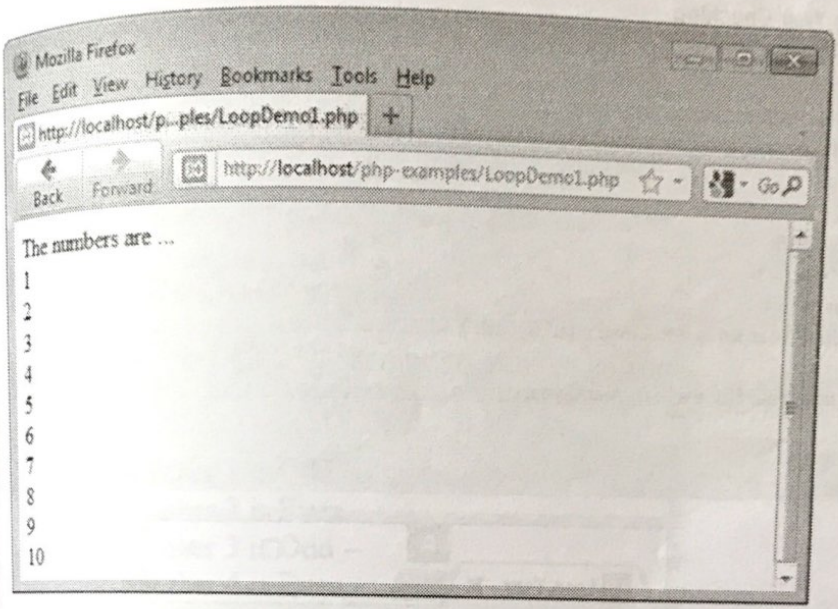
- The **while, for** and **do-while** statements of PHP are similar to JavaScript.
- Following is a simple PHP script which displays the first 10 number

PHP Document[LoopDemo1.php]

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

Scanned with CamScanner

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++;
}
?>
```

Example 4.5.1 *Write PHP programs to*

1. *To print whether current year is leap year or not.*
2. *To print whether given number is odd or even.*

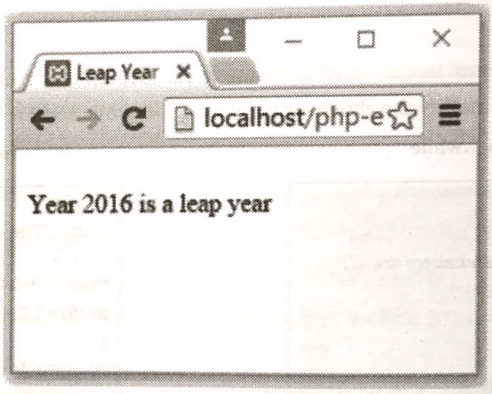
GTU : Winter-11, Marks 7

Scanned with CamScanner

Solution :

1. Leap Year Checking

```
<html>
<head>
<title>Leap Year Demo</title>
</head>
<body>
<?php
 $year=2016;
 print "<br/>";
 if($year%4==1)
  { printf("Year %d is not a leap year",$year); }
 else
  { printf("Year %d is a leap year",$year); }
?>
</body>
</html>
```



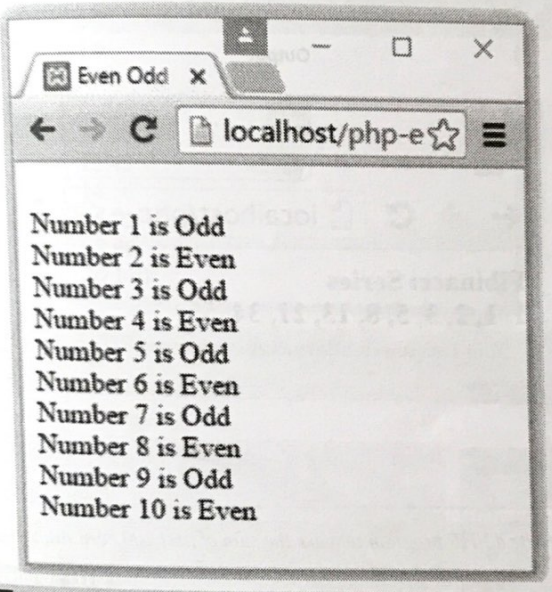
2. First 10 numbers - Even or Odd display

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

Scanned with CamScanner

```
 { printf("Number %d is Odd",$num); }
else
 { printf("Number %d is Even",$num); }
}
?>
</body>
</html>
```

Output



Example 4.5.2 *Write PHP program to print first 10 Fibonacci numbers.*

GTU : Winter-11, Marks 7

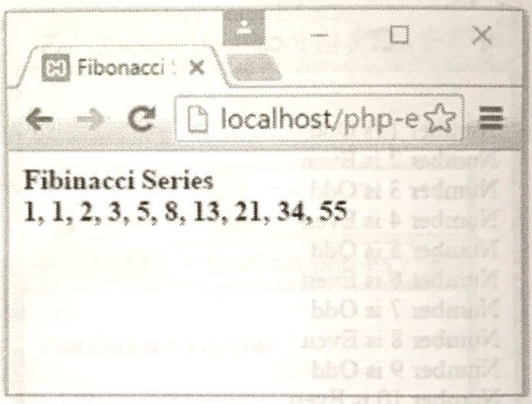
Solution :

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

Scanned with CamScanner

```
for($count=1;$count<9;$count++)
{
        $k=$i+$j;
        $i=$j;
        $j=$k;
        printf(", %d",$k);
}
?>
</body>
</html>
```

Output



Example 4.5.3 *Write a PHP program to make the sum of first 100 even numbers.*

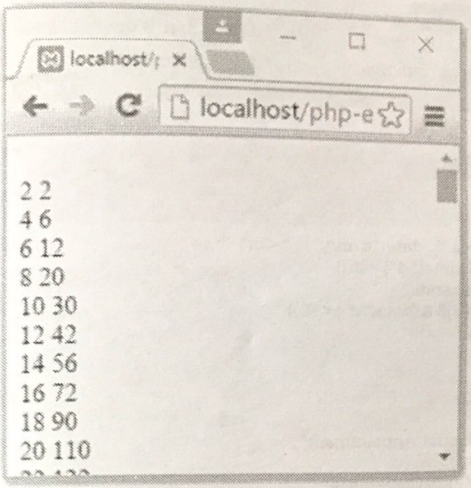
GTU : Winter-12, Marks 7

Solution :

```
<?php
$sum=0;
for($i=1;$i<=200;$i++)  //First 100 odd numbers are between 1 to 200
{
if($i%2==0) //checking odd number
{
 $sum+=$i;
 print "<br/> $i  $sum";
}
}
```

Scanned with CamScanner

Output

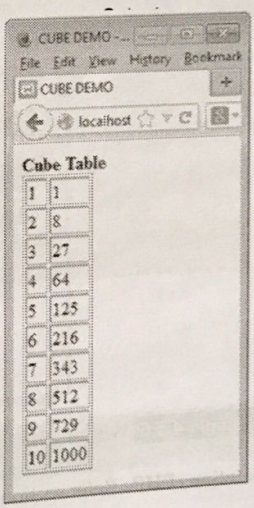


Example 4.5.4 *Write a PHP program to display table of cube for 1 to 10.*

GTU : Summer-13, Marks 7

Solution :

```
<html>
<head>
<title>CUBE DEMO</title>
</head>
<body>
<?php
 print "<b>Cube Table<br/>";
 print "<table border='1'>";
 for($count=1;$count<=10;$count++)
 {
      $result=$count*$count*$count;
   print "<tr><td>$count</td><td>$result</td></tr>";
 }
 print "</table>";
?>
</body>
</html>
```



Scanned with CamScanner

Example 4.5.5 *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.*

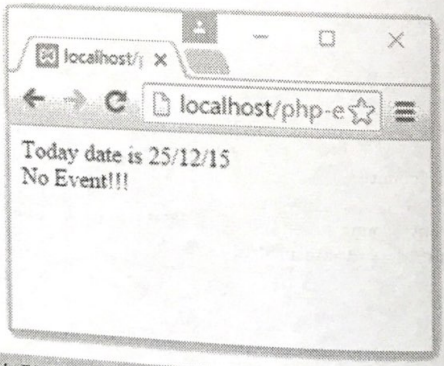
iii) If date is 10, go to conference. *iv) If date is other than 3 and 10, no events are scheduled.*

GTU : Summer-14, Marks 7

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



Example 4.5.6 *What is PHP? Write a PHP program to find whether entered year is leap year or not ?*

GTU : Winter-14, Marks 7

Solution : PHP : Refer section 4.1.1.

Scanned with CamScanner

PHP Document for Leap Year Checking : Refer example 4.5.1.

Example 4.5.7 *Write a PHP script to print following pattern.* **GTU : Summer-16, Marks 7**

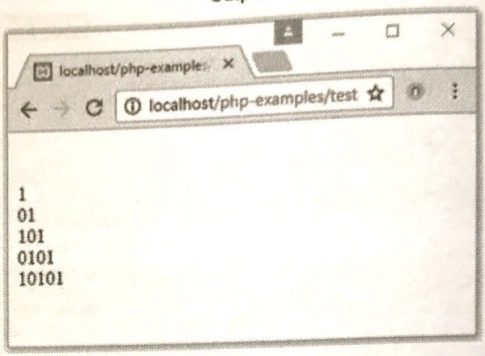
```
1
01
101
0101
10101
```

Solution :

```
<html>
<head>
</head>
<body>
<?php
  for($i=0;$i<7;$i++)
  {
    for($j=1;$j<$i;$j++)
    {
      if(($i+$j)%2==0)
      {
        printf("0");
      }
      else
      {
        printf("1");
      }
    }
    print "<br/>";
  }
?>
</body>
</html>
```

Scanned with CamScanner

Output



4.6 Arrays

GTU : Winter-11,13, Summer-12,16, Marks 7

- 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.

4.6.1 Array Creation

- There are two ways to create an array in PHP.
- The first way is to use the construct **array**. For example -

```
$mylist=array(10,20,30,40,50);
```

- The second way is to assign the value directly to the array. For example -

```
$mylist[0]=10;
```

- An empty array can also be created using the **array** construct. For example -

```
$mylist=array();
```

- We can have mixed type of elements in the array. It is as follows -

```
$mylist=array("Archana" => "Pune", "Shilpa =>89.33);
```

4.6.2 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 -

Scanned with CamScanner

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

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

4.6.3 Types of Arrays in PHP

PHP supports three types of arrays -

- Numeric arrays
- Associative arrays
- Multidimensional arrays

Let us get introduced with these arrays with the help of illustrative examples -

1) Numeric Arrays

This type of array is similar to arrays used in C. The data is placed in the array using the numeric index. For example

```
$student=array("Chitra","Monika","Sagar","Nilesh","Shilpa");
By this assignment we will get
$student[0]="Chitra";
$student[1]="Monika";
$student[2]="Sagar";
$student[3]="Nilesh";
$student[4]="Shilpa";
```

2) Associative Arrays

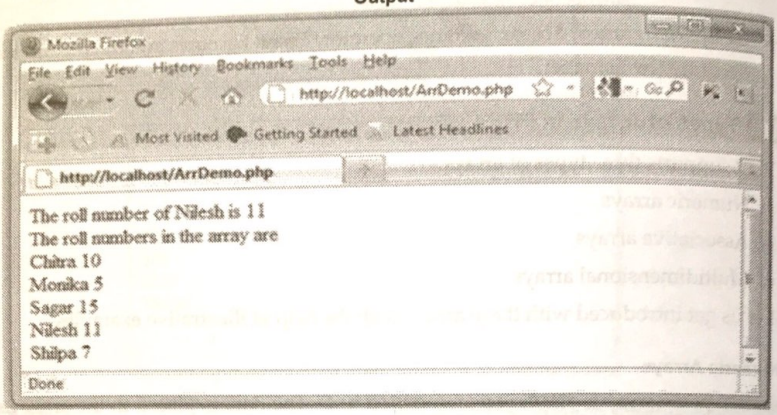
The associative arrays are special type of arrays in which the ID key is associated with value. Following is a simple PHP script which makes use of associative arrays

ArrDemo.php

```
<?php
$student=array("Chitra"=>10,"Monika"=>5,"Sagar"=>15,"Nilesh"=>11,"Shilpa"=>7);
print "The roll number of Nilesh is ".$student['Nilesh'];
print "<br/>The roll numbers in the array are ";
foreach($student as $name=>$value)
print "<br/>".$name." ".$value;
?>
```

Scanned with CamScanner

Output



Using **foreach** loop statement we can display the contents of the array sequentially. The syntax of **foreach** statement is

```
foreach(array_name as scalar_variable)
 loop body
foreach(array_name as key=>value)
 loop body
```

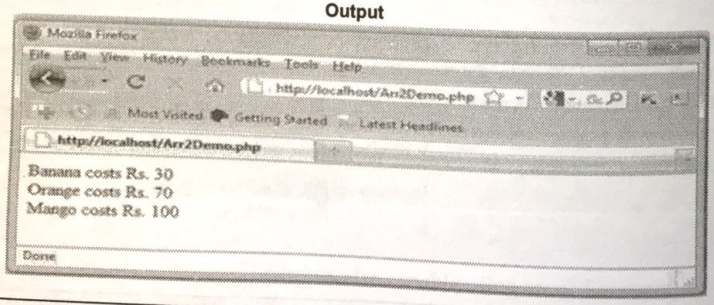
3) Multidimensional Array

The most commonly used type of multidimensional array is two dimensional or three dimensional array.

Arr2Demo.php

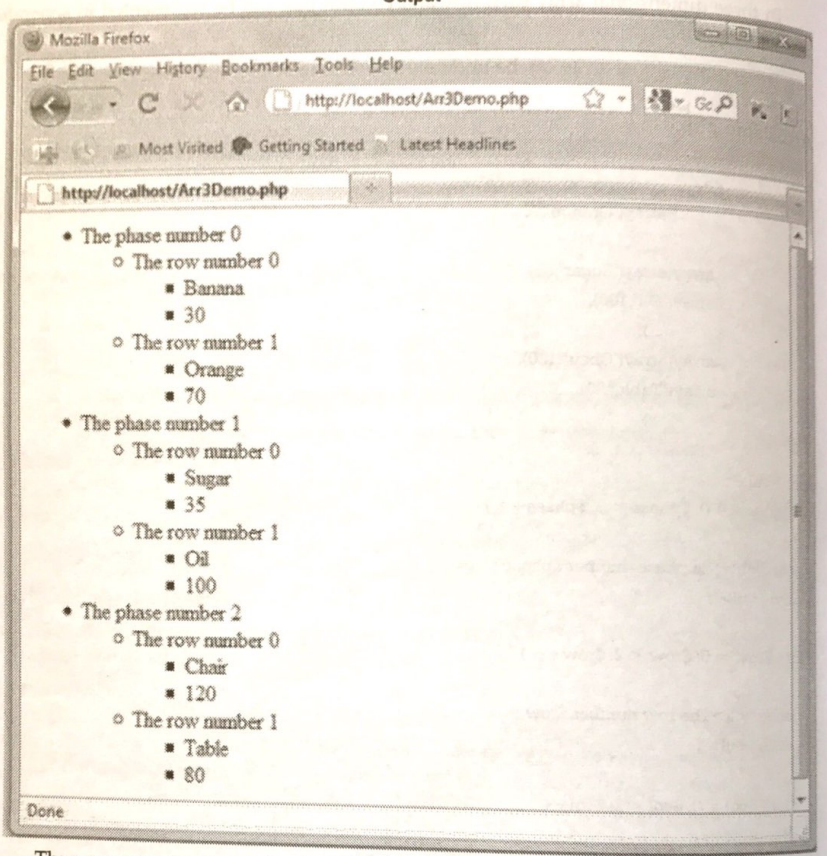
```
<?php
$shopping=array(array("Banana",30),array("Orange",70),array("Mango",100));
echo $shopping[0][0]." costs Rs. ".$shopping[0][1]."<br/>";
echo $shopping[1][0]." costs Rs. ".$shopping[1][1]."<br/>";
echo $shopping[2][0]." costs Rs. ".$shopping[2][1]."<br/>";
?>
```

Output



Scanned with CamScanner

Output



Thus we can go on continuing the multidimensional array by defining one array inside the other

Example 4.6.1 *Use an array to store student information such as enrollmentno, name, semester and percentage. Output the data to a web page using PHP.*

GTU : Winter-13, Marks 7

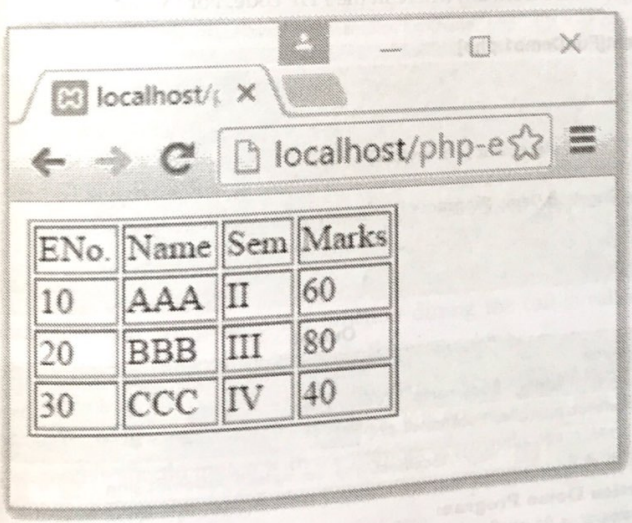
Solution :

```
<html>
<head></head>
<body>
<?php
```


Scanned with CamScanner

```
$a=array(array(10,"AAA","II",60),array(20,"BBB","III",80),array(30,"CCC","IV",40));
echo "<table border='1'>";
echo "<tr>";
echo"<td>ENo.</td><td>Name</td><td>Sem</td><td>    Marks</td>";
echo "</tr>";

for($i=0;$i<3;$i++)
{
 echo "<tr>";
for($j=0;$j<4;$j++)
{
    echo "<td>";
    echo $a[$i][$j];
    echo "</td>";
}
echo "</tr>";
}
echo "</table>";
?>
</body>
</html>
```

Output



ENo.	Name	Sem	Marks
10	AAA	II	60
20	BBB	III	80
30	CCC	IV	40

Scanned with CamScanner

Review Questions

1. *What are the different types of arrays in PHP ? Explain with example to process the arrays in PHP.* GTU : Winter-11, Marks 7
2. *Explain the PHP arrays with example.* GTU : Summer-12, Marks 4
3. *Explain types of array in PHP with example.* GTU : Summer-16, Marks 7

4.7 Functions

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

4.7.1 General Characteristics of Functions

- The syntax of the function definition is as follows -

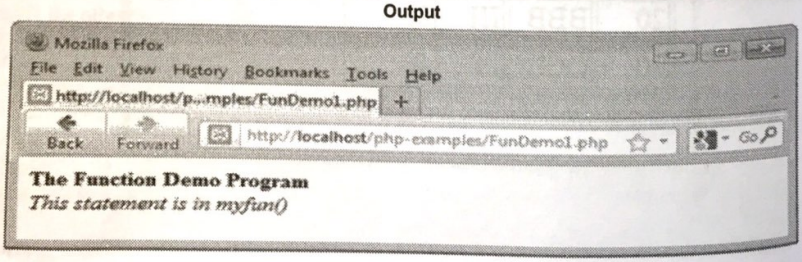
```
function name_of_function(parameter list)
{
 statements to be executed in function
 ...
 ...
 ...}
```

- 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



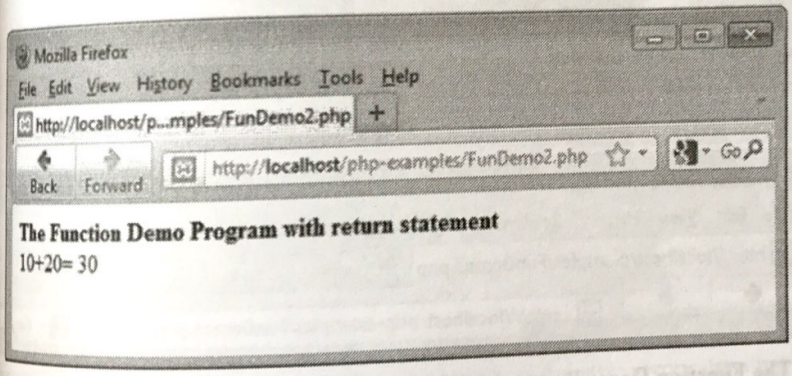
Scanned with CamScanner

- 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



4.7.2 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.

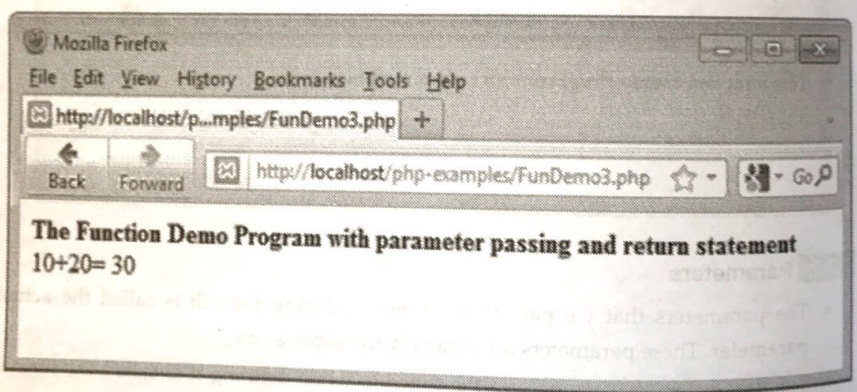
Scanned with CamScanner

- 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



4.8 Browser Control and Detection

GTU : Winter-13, Marks 4

PHP can control various features of browsers. For examples many times there is a need to reload the web page. This can be done with the help of PHP Script using header command.

Scanned with CamScanner

Example 4.8.1 *What PHP can do with header() command ?* **GTU : Winter-13, Marks 3**

Solution : The header() function sends a raw HTTP header to a client. It specifies the header string to send. For example

```
<html>
<?php
header('Location: http://www.google.com/');
?>
```

The main thing to remember with the header function is that it must be the first thing that sends output in your script. If it is not, you will receive the "headers already sent" error. One of the important use if this you can find out the number of times one has visited the particular web page.

PHP can also be useful in detecting the browser on which the script is running. For detecting the browser, the browser_ID can be used. Following command can be used for that purpose

```
$browser_ID=$_SERVER['HTTP_USER_AGENT'];
```

The $_SERVER is a global array using which the current status can be obtained. The HTTP_USER_AGENT is an environment variable which contains the information about the browser.

Example 4.8.2 *Using PHP, how can it be identified that server is dealing with which browser ?* **GTU : Winter-13, Marks 4**

Solution : Following PHP script can be used to indentify that the server is dealing with which browser

```
<?php
echo $_SERVER['HTTP_USER_AGENT'];
?>
```

4.9 String

GTU : Winter-11,13,14, Marks 7

Various operations on string are - finding the length of the string, concatenating the two strings, comparing two strings, reversing the string and so on. PHP has various functions that support these operations. Following is a simple PHP script in which various functions are used to perform string operations.

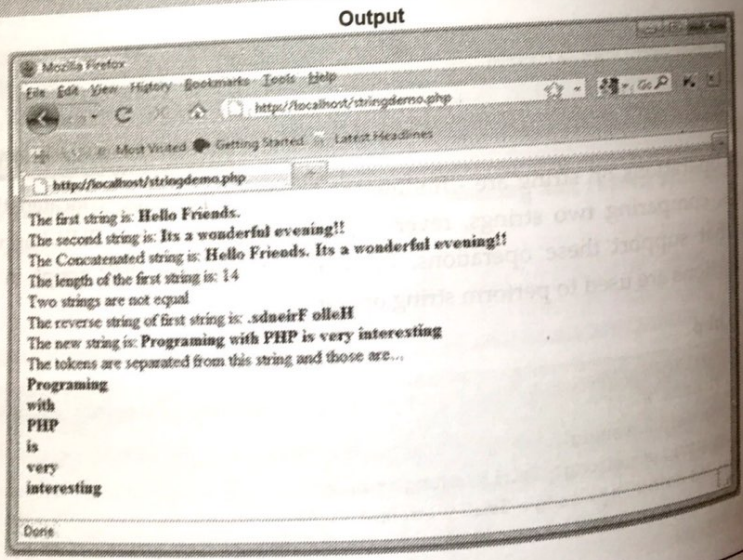
stringdemo.php

```
<?php
$str1="Hello Friends.";
$str2="Its a wonderful evening!!";
print "The first string is:<strong> $str1</strong><br/>";
print "The second string is:<strong> $str2</strong><br/>";
print "The Concatenated string is: <strong>";
print $str1." ".$str2;
```


Scanned with CamScanner

```
print "</strong><br/>";
print "The length of the first string is: ";
print strlen($str1);
print "<br/>";
if(strcmp($str1,$str2)==0)
 print "Two strings are equal";
else
 print "Two strings are not equal";
print "<br/>The reverse string of first string is: <strong>";
print strrev($str1);
print "</strong>";
$str="Programing with PHP is very interesting";
print "<br/>The new string is: <strong>";
print $str;
print "</strong><br/>";
print "The tokens are separated from this string and those are...<br/><strong>";
$t=strtok($str," ");
while($t!=false)
{
 echo "$t<br/>";
 $t=strtok(" ");
}
print "</strong>";

?>
```

Output



Scanned with CamScanner

There are other commonly used functions enlisted in the following table

Function	Description
echo()	It displays the string
trim()	Removes the white spaces from both ends of the string.
ltrim()	Removes the white space characters from the left side of the string.
strops	Returns the position of the first occurrence of a string inside another string
strstr	returns the substring from the first string which is passed as a parameter.

Review Questions

1. *Explain string processing with respect to PHP.* GTU : Winter-11, Marks 7
2. *Explain with suitable example, string manipulation with PHP.* GTU : Winter-13,14, Marks 7

4.10 Form Processing

GTU : Summer-12,13,16, Marks 10

For handling the form basically the PHP code is directly embedded within the XHTML document. The values of the form elements are can be obtained by the PHP script using $_GET and $_POST variables. The HTML makes use of GET or POST method when it invokes the PHP script.

If we collect the values using GET method then $_GET variable is used in the PHP script. Note that if the GET method is used then all variable names and values will be displayed in the URL as a **Query String**. But if we use POST method then these values will not be displayed in the URL. Hence if we send some sensitive information such as password or PIN using the HTML form then we must not use GET method.

Example :

We will build a simple application for online purchase of books.

Step 1 : Create a simple HTML web page which will collect the user information. It is as follows -

MainPage.html

```
<html>
<head>
 <title>ONLINE SHOPPING</title>
</head>
<body bgcolor="khaki">
<form action="http://localhost/myscript.php" method="post"/>
<table>
<tr>
```

Scanned with CamScanner

```
<td><strong>Personal Information:</strong></td>
</tr>
<tr>
 <td>Name:</td>
 <td><input type="text" name="MyName" size=30/></td>
</tr>
<tr>
 <td>Address</td>
 <td><input type="text" name="MyAddress"size=50/></td>
</tr>
<tr>
 <td>Phone</td>
 <td><input type="text" name="MyPhone"size=15/></td>
</tr>
<tr>
<td>Sex:</td>
<td><input type="radio" name="Group1" value="Male"/>Male</td>
<td><input type="radio" name="Group1" value="Female"/>Female</td>
</tr>
<tr>
<td><strong>Book Purchase:</strong></td>
</tr>

 <tr>
  <td>Book Name:</td>
  <td>
   <select name="MyMenu">
   <option value="Software Engineering">Software Engineering</option>
   <option value="Operating System">Operating System</option>
   <option value="Compiler Design">Compiler Design</option>
   <option value="Theory Of Computation">Theory Of Computation</option>
   <option value="Data Structures">Data Structures</option>
   </select>
  </td>
 </tr>
 <tr>
  <td>Price:</td><td><input type="text" name="Price"/></td>
 </tr>
 <tr>
  <td>Quantity:</td>
  <td>
  <select name="Qty">
   <option value="1">1</option>
   <option value="2">2</option>
   <option value="3">3</option>
   <option value="4">4</option>
   <option value="5">5</option>
```


Scanned with CamScanner

```
</select>

</td>
</tr>
<tr>
<td></td><td><input type="submit" value="Submit"/></td>
</tr>
</table>
</body>
</html>
```

Step 2 : Now create a PHP script which will read the values entered by you in step 1, from the MainPage.html. Then this script will compute the total amount of purchase and will display the related information on the web page.

myscript.php

```
<html>
<body bgcolor="lime">
<table border="1">
<tr>
<th>Name</th><th>Address</th><th>Phone</th><th>Book Name</th><th>Amount</th>
</tr>
<tr>
<td>
<?php
print $_POST["MyName"];
?>
</td>
<td>
<?php
print $_POST["MyAddress"];
?>
</td>
<td>
<?php
print $_POST["MyPhone"];
?>
</td>
<td>
<?php
print $_POST["MyMenu"];
?>

</td>
<td>
<?php
```

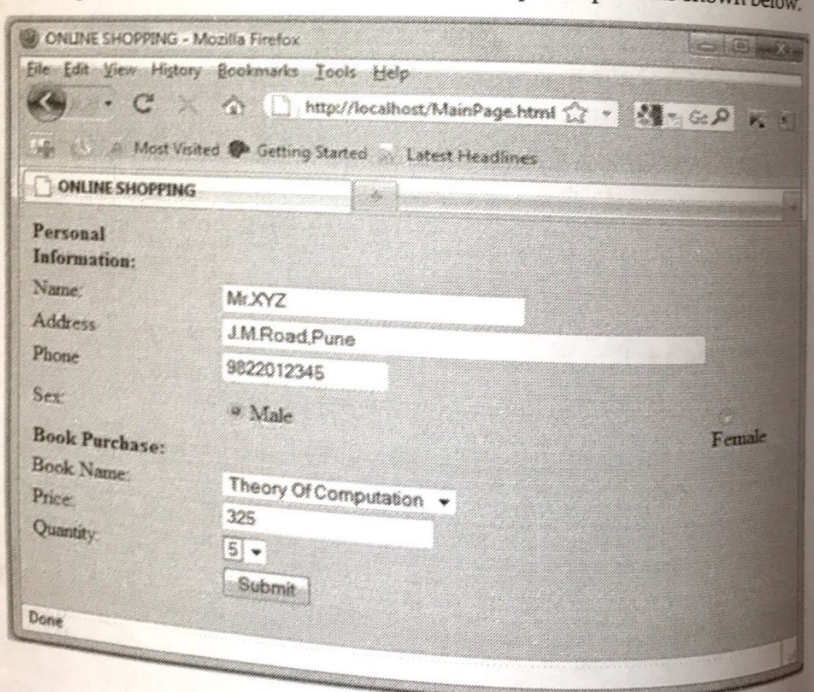
Scanned with CamScanner

```
$val=$_POST["Price"]*$_POST["Qty"];
print $val;
?>
</td>
</tr>
</table>
</body>
</html>
```

Explanation :

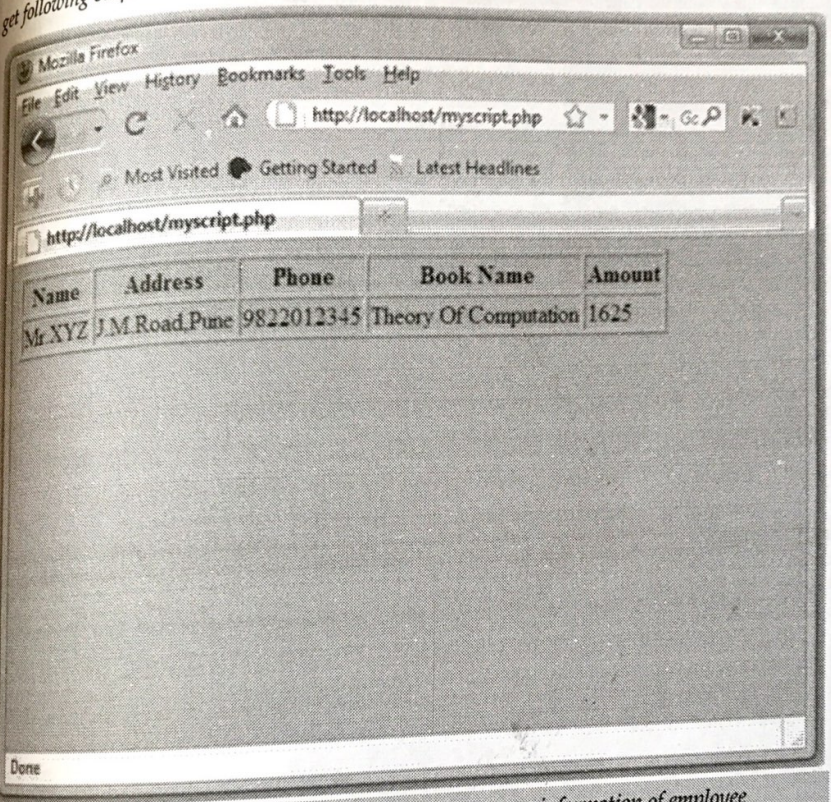
Note that we have used $_POST variable to read the values sent by the HTML form. Note that our **MainPage.html** page invokes the php script **myscript.php** using action element of the form. The method used is POST. While writing the PHP script for handling form, normally the PHP code is embedded within the HTML document.

Step 3 : Open some suitable web browser and open the HTML page created in step 1. Fill up the fields with suitable information. The sample output is as shown below.



Scanned with CamScanner

Click the Submit button. Then the PHP script(created in step 2) will be invoked and you will get following output –



Example 4.10.1 *Write modules using HTML and PHP to store information of employee (employee id, job title, years of experience) in an array. And output the data to a web page by arranging the employees in ascending order of experience.* **GTU : Summer-12, Marks 7**

Solution :

Step 1 : Create an HTML document that reads the record for employees. The code is as below

Input.html

```
<html>
<head>
<title> HTML and PHP Array Demo </title>
</head>
<body>
```


Scanned with CamScanner

```
<form method="post" action="http://localhost/empRecord.php">
<p>Enter Employee ID</p>
 <input name="empID[]" size="5" type="text" />
 <input name="empID[]" size="5" type="text" />
 <input name="empID[]" size="5" type="text" />
 <input name="empID[]" size="5" type="text" />
<p>Enter Job Titles</p>
 <input name="jobTitle[]" size="10" type="text" />
 <input name="jobTitle[]" size="10" type="text" />
 <input name="jobTitle[]" size="10" type="text" />
 <input name="jobTitle[]" size="10" type="text" />
<p>Enter Years of Experience</p>
 <input name="yrsExp[]" size="2" type="text" />
 <input name="yrsExp[]" size="2" type="text" />
 <input name="yrsExp[]" size="2" type="text" />
 <input name="yrsExp[]" size="2" type="text" />
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

Step 2 : Create a PHP document that displays the employees by arranging them in ascending order of experience. Here is the code -

empRecord.php

```
<html>
<body>
<table border="1">
<tr>
<th>Emp ID </th>
<th>Job Title </th>
<th>Years</th>
</tr>
<?php

$empID=$_POST['empID'];
$jobTitle=$_POST['jobTitle'];
$array=$_POST['yrsExp'];

$array = array_map(create_function('$value', 'return (int)$value;'),$array);

$arr = array();
for($i=0;$i<count($empID);$i++)
```

Obtaining the records from HTML page

Converting yrs of experience in integer and storing it in array[]

Scanned with CamScanner

```
{
$arr[$i]['id'] = $empID[$i];
$arr[$i]['title'] = $jobTitle[$i];
$arr[$i]['yrs'] = $array[$i];
}
```

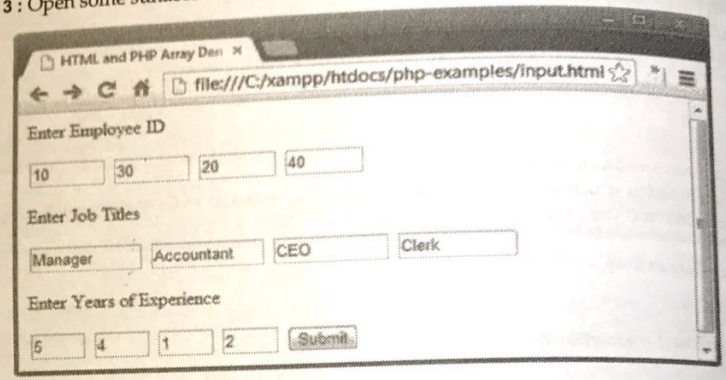
Combining 3 one dimensional arrays in one two dimensional array.

```
$result = sortTwoDArray($arr,'yrs');
//sorting function which sorts the Two dimensional array based on years of experience
function sortTwoDArray($arr, $arrKey, $sortOrder=SORT_ASC)
{
foreach ($arr as $key => $row)
{

$key_arr[$key] = $row[$arrKey];
}
array_multisort($key_arr, $sortOrder, $arr);
return $arr;
}
//The result array contains the sorted records
for($i=0;$i<count($empID);$i++)
{
 print"<tr>";
 print"<td>";
 echo $result[$i]['id'];
 print"</td>";
 print"<td>";
 echo $result[$i]['title'];
 print"</td>";
 print"<td>";
 echo $result[$i]['yrs'];
 print"</td>";
 print"</tr>";
}
?>
</table>
</body>
</html>
```

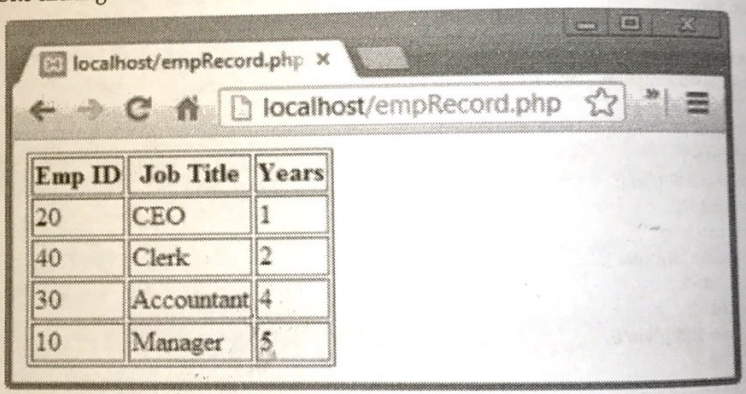
Displaying output in tabular form by arranging Employees in ascending order of experience.

Scanned with CamScanner

Step 3 : Open some suitable web browser and get the output as follows -



One clicking the Submit button we will get -



Emp ID	Job Title	Years
20	CEO	1
40	Clerk	2
30	Accountant	4
10	Manager	5

Example 4.10.2 *Write a PHP program that receives the value of N using HTML form and displays the first N Fibonacci numbers as HTML list.* **GTU : Summer-12, Marks 10**

Solution :

Step 1 : We will create an HTML document that sends the value of N to PHP program. It is as given below -

input.html

```
<html>
<head>
<title> HTML and PHP Fibonacci Demo </title>
</head>
<body>
```


Scanned with CamScanner

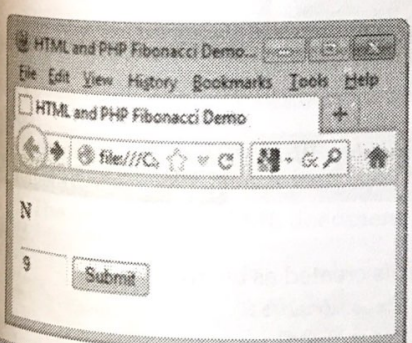
```
<form method="post" action="http://localhost/DisplayFib.php">
<p>N</p>
<input name="Num" size="3" type="text" />
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

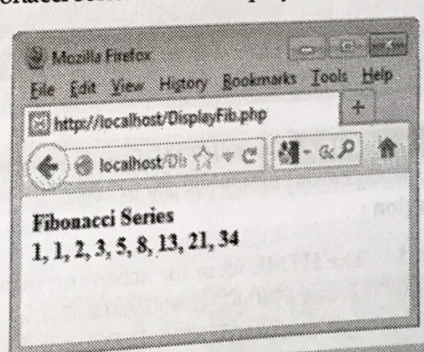
Step 2 : Now the code for the PHP document will be

DisplayFib.php

```
<html>
<body>
<?php
 $i=1;
 $j=1;
 $val=$_POST["Num"];
 print "<b>Fibonacci Series<br/>";
 printf("%d, %d",$i,$j);
for($count=1;$count<$val-1;$count++)
 {
      $k=$i+$j;
      $i=$j;
      $j=$k;
      printf(", %d",$k);
 }
?>
</body>
</html>
```

Step 3 : Open the html document in web browser and click on Submit button the PHP program will be invoked and the Fibonacci series will be displayed.





Example 4.10.3 *Create HTML form with one textbox to get user's name. Also write PHP code to show length of entered name when, the HTML form is submitted.*

GTU : Summer-13, Marks 7

Scanned with CamScanner

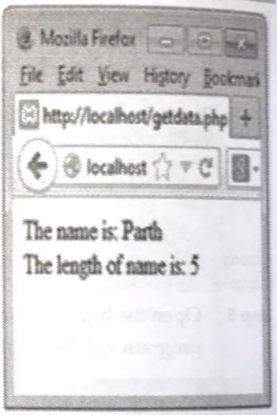
Solution :

Step 1 : The HTML form can be created as follows

```
<!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" xml:lang="en" lang="en">
<head><title> HTML-PHP DEMO</title>
</head>
<body>
  <form method="post" action="http://localhost/getdata.php">
      Name: <input type="text" name="myname" size="20"/>
      <br/>
      <input type="submit" name="submit" value="submit"/>
  </form>

</body>
</html>
```

Step 2 : The PHP script to display the length of submitted name is as written below

```
<?php
print "The name is: ";
print $_POST["myname"];
$len= strlen($_POST["myname"]);
print "<br/> The length of name is: ";
print $len;
?>
```



Example 4.10.4 *Create HTML form to enter one number. Write PHP code to display the message about number is odd or even.* **GTU : Summer-13, Marks 7**

Solution :

Step 1 : The HTML form for accepting number is created as below -

```
<!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" xml:lang="en" lang="en">
<head><title> HTML-PHP DEMO</title>
</head>
<body>
```

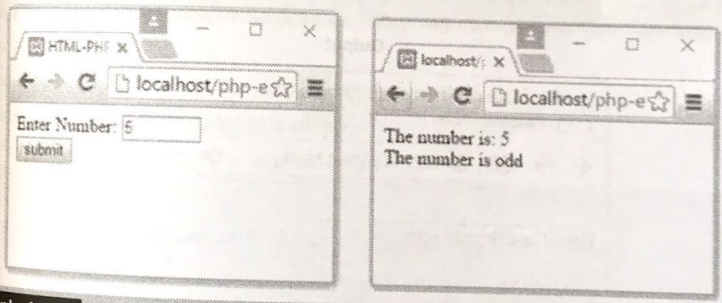
Scanned with CamScanner

```
    <form method="post" action="http://localhost/php-examples/getdata.php">
        Enter Number: <input type="text" name="mynum" size="5"/>
        <br/>
        <input type="submit" name="submit" value="submit"/>
    </form>
</body>
</html>
```

Step 2 : The PHP script deciding whether the number is even or odd is as given below -

```
<?php
print "The number is: ";
print $_POST["mynum"];
$a= $_POST["mynum"];
if($a%2==1)
 print "<br/> The number is odd ";
else
 print "<br/> The number is even ";
?>
```

Output



Example 4.10.5 *Write a PHP script that validate email address using Regular Expression when user click on VALIDATE button.* **GTU : Summer-16, Marks 7**

Solution :

Step 1 : Create an input HTML form using which the email ID is accepted from the user. The code for this HTML document is as follows

```
<html>
<head>
  <title>Validation Form</title>
</head>
<body>
<form id="contact_form" method="post" action="http://localhost/php-examples/ValidateEmail.php">
<br />
 Email <input type="text" name="email" class="textfield" value="" /> <br />
```


Scanned with CamScanner

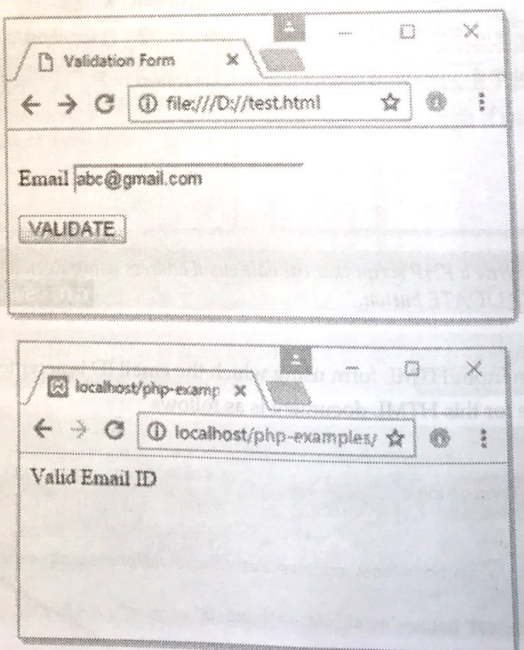
```
    <p><input type="submit" name="submit" class="button" value="VALIDATE" /></p>
</form>
</body>
</html>
```

Step 2 : Create a PHP script for form validation. This form accepts the email id sent by the above HTML from and checks it validity

ValidateEmail.php

```
<?php
$reg = '/^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*(\.[a-z]{2,3})$/';
$email=$_POST["email"];
if (preg_match($reg, $email))
{
    echo "Valid Email ID";
}
else
{
    echo "Invalid Email ID.";
}
?>
```

Output



Scanned with CamScanner

7.1 Asynchronous Web Requests using AJAX

7.1.1 Introduction to AJAX

- AJAX is a Asynchronous **Java Script** and XML
- Here
  - **Asynchronous** means, the execution of script does not disturb the user's work.
  - **JavaScript** because, it makes use of Javascripting to do the actual work.
  - **XML** because along with JavaScript the XML is also supported to perform the given task in AJAX.
- It is not a new programming language but it is a kind of web document which adopts certain standards.
- AJAX allows the developer to exchange the data with the server and updates the part of web document without reloading the web page.

7.1.2 Architecture

- When user makes a request, the browser creates a object for the **HttpRequest** and a request is made to the server over an internet.
- The Server processes this request and sends the required data to the browser.
- At the browser side the returned data is processed using JavaScript and the web document gets updated accordingly by sending the appropriate response.

Following Fig. 7.1.1 illustrates this working.

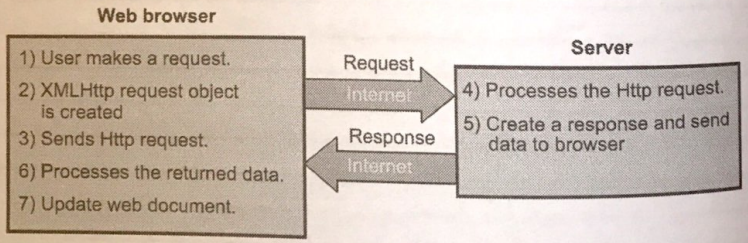


Fig. 7.1.1 Working of AJAX

Scanned with CamScanner

Similarities and Differences

Sr.No.	Traditional Web Application Architecture	Ajax Based Web Application Architecture
1.	The web applications architectures are based on HTTP request response protocol.	
2.	At the client side the browser client has only one component and that is – user interface.	At the client side the browser client has user interface and AJAX Engine due to which the client gets quick response from the server.
3.	The architecture is based on client server communication.	
4.	It makes use of synchronous communication. The client has to wait to get the response from the server then only client can make the request to the server. It is start-stop-start-stop kind of communication.	By adding a new layer of AJAX Engine at the client side, it eliminates the start-stop-start-stop nature of communication and makes the communication asynchronous.
5.	With traditional web application architecture user cannot get rich user interface experience.	With AJAX architecture user gets rich user interface experience.
6.	It is less responsive.	It is more responsive.
7.	Building web application is simple.	The development time is more while designing the Ajax based web application.
8.	Limited use of JavaScript, CSS and XML technologies. In fact - Most user actions in the interface trigger an HTTP request back to a web server.	Extensive use of JavaScript, CSS and XML - i) standards-based presentation using XHTML and CSS; ii) Dynamic display and interaction using the Document Object Model(DOM); iii) Data interchange and manipulation using XML and XSLT; iv) Asynchronous data retrieval using XMLHttpRequest; v) JavaScript binding everything together

Scanned with CamScanner

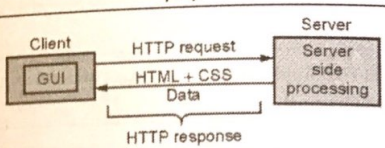


Fig. 7.1.2 Traditional web application architecture

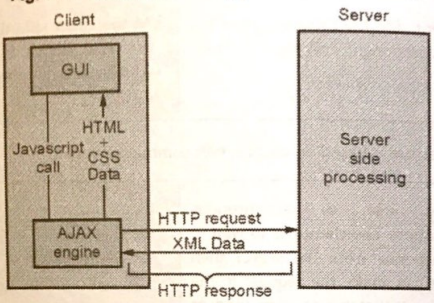


Fig. 7.1.3 AJAX web application architecture

Merits of AJAX

1. Response time is faster, hence increases performance and speed.
2. XMLHttpRequest object call as a asynchronous HTTP request to the Server for transferring data both side. It's used for making requests to the non-Ajax pages.
3. It is used in form validation.
4. It performs fetching of data from database and storing of data into database without reloading page.

Demerits of AJAX

1. It has browsing compatibility issues.
2. It is impossible to bookmark AJAX updated page contents.
3. Search engine would not crawl AJAX generated content. Hence, Search Engines like Google cannot index AJAX pages.

7.1.3 XMLHttpRequest Object

- XMLHttpRequest object is an important element in AJAX.
- XMLHttpRequest is an API which is used by JavaScript, VBScript and some other scripting languages.

Scanned with CamScanner

- The methods of XMLHttpRequest are used in transferring data between a Web browser and a web server.
- It is because of XMLHttpRequest object, to update parts of web page without reloading the whole page.
- **Features of XMLHttpRequest**
  1. **Requests the data** from the server after the page gets loaded.
  2. **Receives the data** from the server after the page gets loaded.
  3. **Sends the data** to the server behind the scene.
  4. **Update** the web page without reloading the page.

Syntax

- Variable = new XMLHttpRequest();

Various Methods of XMLHttpRequest object

Method	Purpose
new XMLHttpRequest()	Creates new XMLHttpRequest object.
Open(method, URL, async,user,pwd)	To make request this method is used. method: The request can be GET or POST URL: The location of file async: true(for asynchronous) and false(synchronous) user: user name(optional) pwd:password(optional)
send()	sends the request to the server.
send(string)	sends the request to the server.
setRequestHeader()	Adds the label- value pair to the header.

Various Properties of XMLHttpRequest object

Property	Description
readyState	Specifies the ready state of XMLHttpRequest 0: request not initialized 1: server connection established 2: request received 3: processing request

Scanned with CamScanner

	4: request finished and response is ready
onreadystatechange	When readystate property changes the onreadystatechange event listener will be automatically invoked
responseText	Returns the response data as a string
status	Returns the status number. Following is the meaning of various numbers returned by the status property 200: "OK" 403: "Forbidden" 404: "Not Found"
statusText	Returns the status text as "OK", or "Forbidden"

Example 7.1.1 *Create an XMLHttpRequest to retrieve data from an XML file and display the data in an HTML table. The data to be retrieved is a collection of stationary items stored in an XML file.*

Solution : Step 1 : Create an xml file storing the stationary items. The code is as follows -

```
<ELEMENTS>
<STATIONARY>
<ITEM>Notebook</ITEM>
<PRICE>40</PRICE>
</STATIONARY>
<STATIONARY>
<ITEM>Punching Machine</ITEM>
<PRICE>60</PRICE>
</STATIONARY>
<STATIONARY>
<ITEM>Pencil Box</ITEM>
<PRICE>40</PRICE>
</STATIONARY>
<STATIONARY>
<ITEM>Eraser</ITEM>
<PRICE>5</PRICE>
</STATIONARY>
</ELEMENTS>
```

Step 2 : The XML file displaying the stationary items in a tabular form is as given below –

test.html

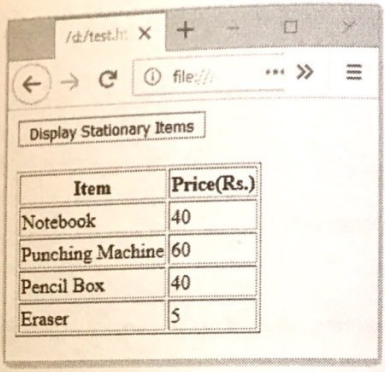
```
<!DOCTYPE html>
```


Scanned with CamScanner

```
<html>
<body>
<button type="button" onclick="loadXMLDoc()">Display Stationary Items</button>
<br><br>
<table id="demo" border="1"></table>
<script>
function loadXMLDoc()
{
 if(window.XMLHttpRequest)
 {
    xmlhttp = new XMLHttpRequest();
 }
     else
{
    xmlhttp = new ActiveXObject("Microsoft.req");//to execute on Inter. Explorer
 }
 xmlhttp.onreadystatechange = function()
 {
     if (this.readyState == 4 && this.status == 200) {
        myFunction(this);
 }
 };
        xmlhttp.open("GET","stationary.xml",true);
    xmlhttp.send();
}
function myFunction(xml)
{
 var i;
     var xmlDoc = xml.responseXML;
      var table="<tr><th>Item</th><th>Price(Rs.)</th></tr>";
     var x = xmlDoc.getElementsByTagName("STATIONARY");
     for (i = 0; i <x.length; i++)
      {
    table += "<tr><td>" +
    x[i].getElementsByTagName("ITEM")[0].childNodes[0].nodeValue +
    "</td><td>" +
    x[i].getElementsByTagName("PRICE")[0].childNodes[0].nodeValue +
    "</td></tr>";
 }
 document.getElementById("demo").innerHTML = table;
}
</script>
</body>
</html>
```


Scanned with CamScanner

Output



7.1.4 Call Back Methods

- Callback functions are used to handle responses from the server in Ajax. A callback function can be either a named function or an anonymous function.
- We can set the value of event handler onreadystatechange equal to anonymous function as follows –

```
req.onreadystatechange=function()
{
    if (req.readyState==4 && req.status==200)
    {
      document.getElementById("myID").innerHTML=req.responseText;
    }
}
```

- We can write the call back function as named function using following steps –

Step 1 : Create XMLHttpRequest obect in global space

```
var req = new XMLHttpRequest();
```

Step 2 : Write a function which acts as a callback function

```
function myCallBkFun()
{
    if (req.readyState==4 && req.status==200)
      {
      Cosole.log(req.responseText);
      }
}
```

Scanned with CamScanner

Step 3 : Now we can set the value of onreadystatechange property to the name of the call back function.

```
//assigning call back function to onreadystatechange property
req. onreadystatechange=myCallBkFun;
```

7.1.5 Coding AJAX Script

Let understand how AJAX works with the help of programming example

```
<html>
 <head>
  <script type="text/javascript">
  function MyFun()
  {
       if (window.XMLHttpRequest)
    {
      req=new XMLHttpRequest();
    }
    else
     {
       req=new ActiveXObject("Microsoft.req");
     }
    req.onreadystatechange=function()
     {
       if (req.readyState==4 && req.status==200)
        {
            document.getElementById("myID").innerHTML=req.responseText;
        }
     }
     req.open("GET","newdata.txt",true);
     req.send();
   }
  </script>
</head>
<body>
<div id="myID">This text can be changed</div>

<button type="button" onclick="MyFun()">Change</button>
</body>
</html>
```

2 3 4 5 1

Script Explanation (Above numbered steps are explained here)

In above script, we have written some text which can be replaced by some another text on button click.

Scanned with CamScanner

1. On button click a function **MyFun** is invoked. Thus client triggers the event.
2. **XMLHttpRequest** object is used to exchange data with a server. This object allows the user to change/update the parts of the web page without reloading it fully. The modern web browsers such as IE7+, FireFox,Chrome have built in **XMLHttpRequest** but old web browsers make use of **ActiveXObject**.
3. When a request to a server is sent, then **onreadystatechange** event is triggered.

The **readyState** property holds the status of the XMLHttpRequest. The **readystate=4** means request is finished and response is ready. The **status = 200** means "OK"

The DOM is modified and response is modified using the statement

```
document.getElementById("myID").innerHTML=req.responseText;
```

4. The request can be sent to the server by using two functions **open()** and **send()**.

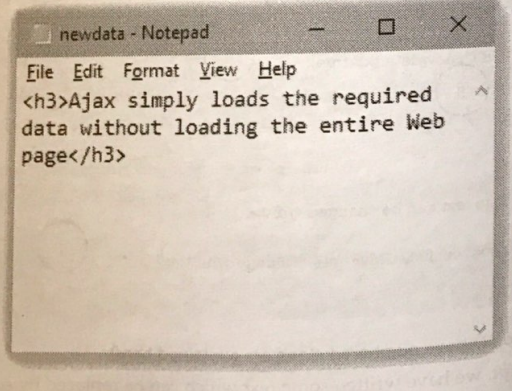
```
req.open("GET","newdata.txt",true);
```

- GET or POST method
- Location of file on server
- true: means asynchronous
  false : means asynchronous

Asynchronous communication allows fast processing of the data.

The **newdata.txt** file contains some updating text using which our web page can be updated.

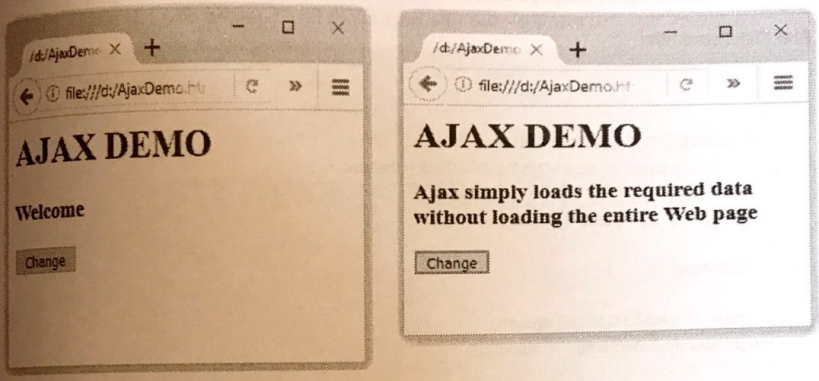
newdata.txt



Scanned with CamScanner

5. The **send()** method sends the request to the server.

Output



Example 7.1.2 *Write AJAX script to obtain the student information stored in XML document. The information should be displayed on clicking the button. It should be displayed in tabular form.*

Sol. :

Step 1 : Create an XML file for storing the student information. The XML file is as follows

Student.xml

```
<Student>
   <student_data>
      <Name>AAA</Name>
      <Marks>45</Marks>
   </student_data>
   <student_data>
      <Name>BBB</Name>
      <Marks>55</Marks>
   </student_data>
   <student_data>
      <Name>CCC</Name>
      <Marks>67</Marks>
   </student_data>
   <student_data>
      <Name>DDD</Name>
      <Marks>84</Marks>
```


Scanned with CamScanner

```
    </student_data>
</Student>
```

Step 2 : Create a AJAX script as follows –

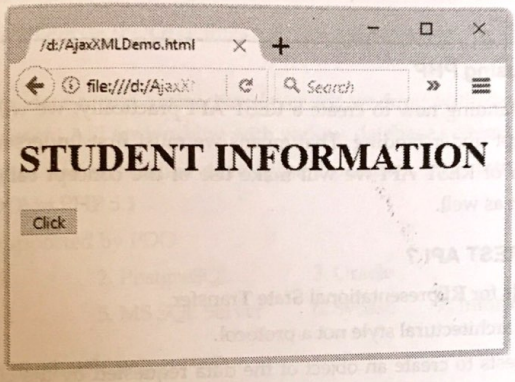
AjaxXMLDemo.html

```
<!DOCTYPE html>
<html>
<body>
<h1>STUDENT INFORMATION</h1>
<button type="button" onclick="MyFun()">Click</button>
<br><br>
<table border="1" id="demo"></table>
<script>
function MyFun()
{
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function()
    {
        if (this.readyState == 4 && this.status == 200)
        {
          Load_XML_File(this);
        }
    };
    xhttp.open("GET", "Student.xml", true);
    xhttp.send();
}
function Load_XML_File(xml)
{
    var i;
    var xmlDoc = xml.responseXML;
    var table="<tr><th>Name</th><th>Marks</th></tr>";
    var x = xmlDoc.getElementsByTagName("student_data");
    for (i = 0; i <x.length; i++)
    {
        table += "<tr><td>" +
        x[i].getElementsByTagName("Name")[0].childNodes[0].nodeValue +
        "</td><td>" +
        x[i].getElementsByTagName("Marks")[0].childNodes[0].nodeValue +
        "</td></tr>";
    }
    document.getElementById("demo").innerHTML = table;
}
```

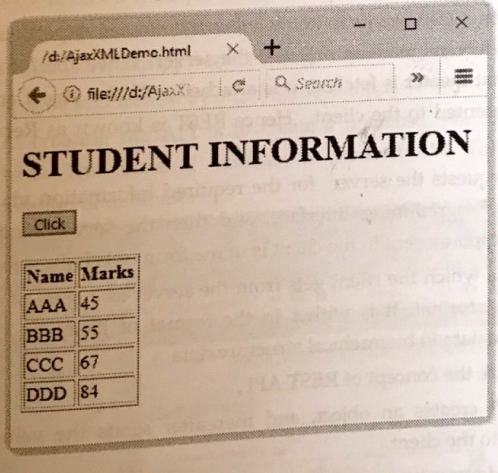

Scanned with CamScanner

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

Step 3 : Open the web browser and the output will be as follows –



On clicking the button we will get



Review Questions

1. *Discuss Ajax client server architecture in detail.*
2. *Compare and contrast the traditional web application architecture and AJAX based web application architecture.*

Scanned with CamScanner

3. *Explain the Ajax client server architecture in detail with a diagram.*
4. *List the pros and cons of AJAX.*
5. *Explain the XMLHttpRequest object methods.*
6. *Explain about the object that helps AJAX reload parts of a web page without reloading the whole page.*

7.2 REST API using PHP

Before understanding how to create a REST API practically, we will just go through some basic concept understanding. That means we must first understand what is REST API. For creation of REST API we will make use of the concept called PDO. We will discuss about that as well.

7.2.1 What is REST API ?

- REST stands for **RE**presentational **S**tate **T**ransfer.
- REST is an architectural style not a protocol.
- REST suggests to create an object of the data requested by the client and send the values of the object in response to the user.
- For example – a User wants to know about the movies at particular theater in your city. Then you can create an object at the server. When the information about the movies in that theater is fetched, it is attached with the object and the state of that object is presented to the client. Hence REST is known as Representational State Transfer.
- The client requests the server for the required information via an API that is - Application Programming Interface, and then the server sends response to the client. The response sent to the client is in the form of HTML web page.
- The response which the client gets from the server as a state of object(REST) is not in structured format. It is wither in the format of JSON or XML. These format represent the data in hierarchical structure data.
- To summarize the concept of REST API ,
  - o REST API creates an object, and thereafter sends the values of an object in response to the client.
  - o It breaks down a transaction in order to create small modules.
  - o Now, each of these modules is used to address a specific part of the transaction. This approach provides more flexibility.

Scanned with CamScanner

- The methods of REST API are based on CURD operations(Create, Read, Update and Delete).

7.2.2 What PDO ?

- The PHP Data Objects (PDO) defines a lightweight interface for accessing databases in PHP.
- It provides a data-access abstraction layer for working with databases in PHP. It defines consistent API for working with various database systems.
- To standardize and streamline development practices, PHP introduced PHP Data Objects (PDO) in PHP 5.1
- Databases supported by PDO

1. MySQL 2. PostgreSQL 3. Oracle
4. Firebird 5. MS SQL Server 6. Sybase 7. Informix

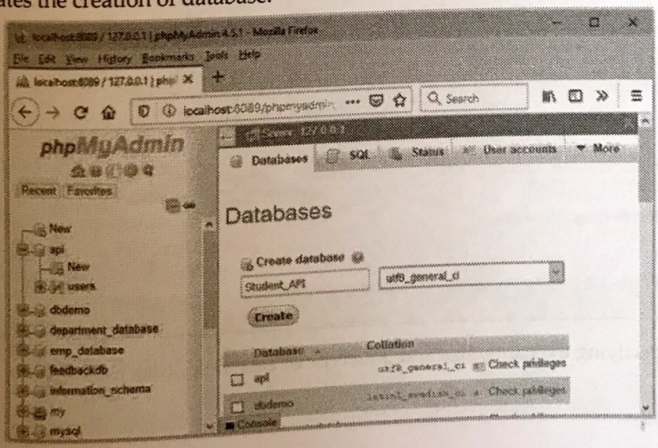
7.2.3 Creating REST API using PHP

In this section we will create a simple REST API for students data.

Prerequisite : XAMPP must be installed. Note that the PHP version must be above 5.1 in this package. The latest xampp version contains such higher version PHP.

Step 1 : Start the apache and tomcat by starting the xampp.

The first step of creating the API is to create a database. We can create a simple database for students. The name of database is **Student_API**. Following screenshot demonstrates the creation of database.



Scanned with CamScanner

```
header('Content-Type: application/json');
echo $API->Display();
?>
```

Script Explanation:

(1) In REST API we have to create an objct. Hence we use a class for which the object can be created.

(2) The class named **API** is created in which the member function **Display** is defined. In this function we first connect to the database.

(3) Using the prepared statement, we execute the query for reading the contents of the database. The query is as follows –

```
Select * from students order by rollno;
```

That means, display the contents of the table named **students** which is arranged in ascending order of rollno.

(4) After execution of this query the data is fetched and stored in **students** array.

(5) This data is encoded in JSON format for display.

(6) The call to this **Display** function is made using the object **$API** of the class.

Step 3(b) : Creation of config.php file

```
<?php
class Connect extends PDO
{
    public function __construct()
    {
        parent::__construct("mysql:host=localhost;dbname=Student_API",'root','',
        array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8"));
        $this->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $this->setAttribute(PDO::ATTR_EMULATE_PREPARES,false);
    }
}
?>
```

Script Explanation :

(1) A database connection is always required to interact with the database. So, we need to know the identifier to access database, i.e., location of the database, database name, username, and password.

(2) A **PDOException** object will be thrown if there is any connection error. We may catch the exception if we want to handle the error condition, or we can also leave it to global exception handler which can be set up by **set_exception_handler()** function.

Scanned with CamScanner

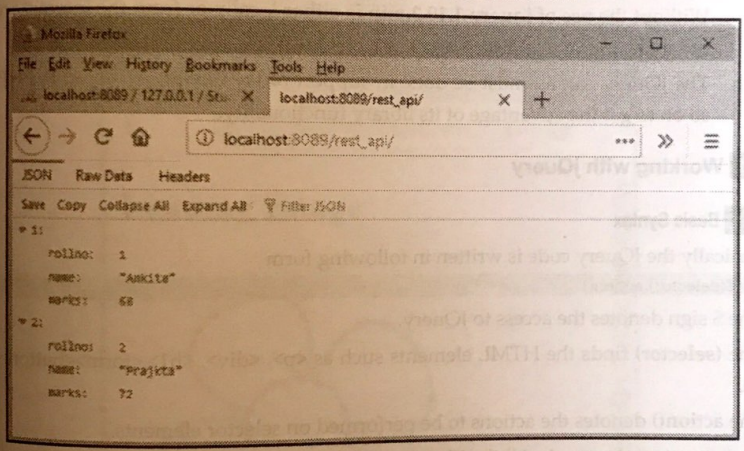
(3) In above script we set the attributes of PDO using **setAttribute** function. The syntax is

```
public PDO::setAttribute ( int $attribute , mixed $value ) : bool
```

Sets an attribute on the database handle.The **PDO::ATTR_ERRMODE** is for **Error reporting** and **PDO::ERRMODE_EXCEPTION** is throws exceptions.

(4) **PDO::ATTR_EMULATE_PREPARES** enables or disables emulation of prepared statements. Some drivers do not support native prepared statements or have limited support for them. Use this setting to force PDO to either always emulate prepared statements (if TRUE and emulated prepares are supported by the driver), or to try to use native prepared statements (if FALSE).

Step 4 : The output can be checked on the web browser as follows -



7.3 JQuery

- JQuery is a fast and concise library created by **John Rosig** in 2006. The moto of this web document is **Write less do more.**
- This scripting is very easy to learn. The JQuery is actually a **Javascript library.**
- JQuery runs almost all the browsers.
- It adds up many advanced, cross-browser functions to standard

Advantages of using JQuery

1. It works on almost all the platforms- This is called as **cross platform compatibility.**

Scanned with CamScanner

2. It has large and advanced set of functionalities.
3. It is **lightweight** about 19 kB in size.
4. It **supports AJAX** technology.
5. It offers **event handling** to capture a wide variety of events such as button clicks, mouse move and so on.
6. It supports plenty of built in **animation effects** to make the web document live and eye catching.
7. It supports **DOM manipulations**. That means it is easy to select DOM elements, traverse them and modifying their content using JQuery.

Disadvantages of using JQuery

1. Without the use of **jquery-1.10.2.min.js** either locally or from the internet we can not make use of the library functions of JQuery.
2. The JQuery coding must be done in support of JavaScript, AJAX, ASP, PHP and so on to get the advantage of its library functionality.

7.3.1 Working with jQuery

7.3.1.1 Basic Syntax

Basically the JQuery code is written in following form

```
$(selector).action()
```

The $ sign denotes the access to JQuery.

The **(selector)** finds the HTML elements such as <p>, , <h1>,<form><button> and so on.

The **action()** denotes the actions to be performed on selector elements.

Document ready event : All the jQuery methods are written in document ready event.

7.3.1.2 Loading JQuery

- The load() method loads data from a server and puts the returned data into the selected element.
- **Syntax**

```
$(selector).load()
```

- **Example**

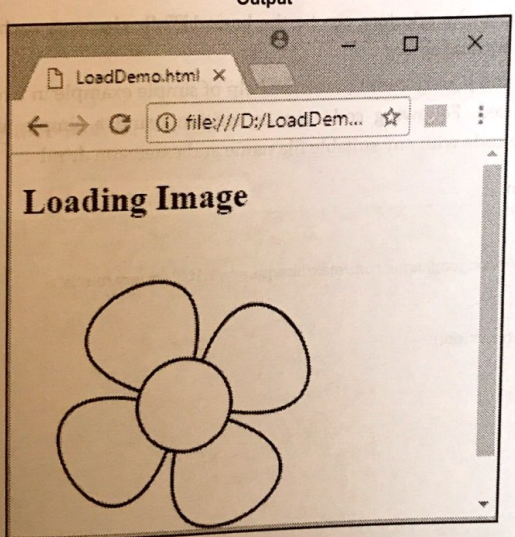
JQUERY Document[LoadDemo.html]

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


Scanned with CamScanner

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<script>
$(document).ready(function()
{
  $("img").load();
});
</script>
</head>
<body>
<h2> Loading Image</h2>
<img src="img1.jpg" alt="flower">
</body>
</html>
```

Output



We can load some external file using the load function for that purpose we need to pass URL to the load function.

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
```


Scanned with CamScanner

```
$(document).ready(function(){
  $("button").click(function(){
    $("#div1").load("input.html");
  });
});
</script>
</head>
<body>
<div id="div1"><h2>Loading File</h2></div>
<button>Click to load file</button>
</body>
</html>
```

7.3.1.3 Selecting Elements

1. The element selector

As we know that, the selectors can be the basic HTML elements such as<p>, , <h1>,<form><button> and so on.

Let us understand JQuery coding with the help of simple example in which the HTML element <p> is used. Following code can be written using a simple text editor like notepad. Save the code with any suitable file name and extension **.html**

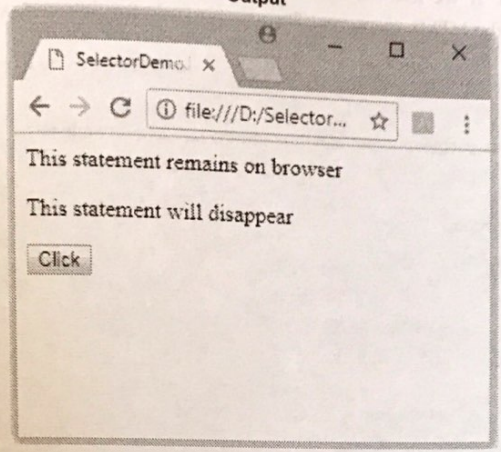
SelectorDemo.html

```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script>
$(document).ready(function()
{
 $("button").click(
function()
 {
  $("p").hide();
 }
 );
});
</script>
</head>

<body>
<div>This statement is remains on browser</div>
<p>This statement will disappear</p>
<button>Click</button>
```

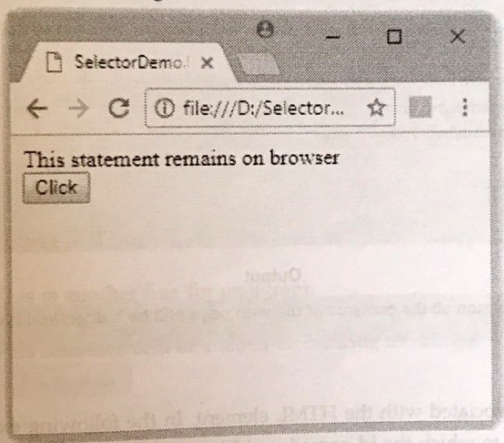

Scanned with CamScanner

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

Output



On clicking the button we will get



Script Explanation :

In the above code, within the **ready** function the JQuery code is written.

For the button element selector the action is **click**. This action invokes the element and the action associated with this element is **hide**. This means that when user clicks the **button** the text written within the will be hidden or disappeared.

Scanned with CamScanner

2. Use of * as a selector

The * means all. If we include selector as * then the corresponding action will be associated with all the HTML elements. The sample code for demonstrating this action is as follows –

```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script>
$(document).ready(function()
{
 $("button").click(
function()
  {
  $("*").hide();
  }
 );
});
</script>
</head>

<body>
<h1> This is a header </h1>
<div>This second statement</div>
<p>This is another statement</p>

<button>Click</button>
</body>
</html>
```

Output

On clicking the click button all the contents of the web page will be hidden and black document will be displayed.

3. The id selector

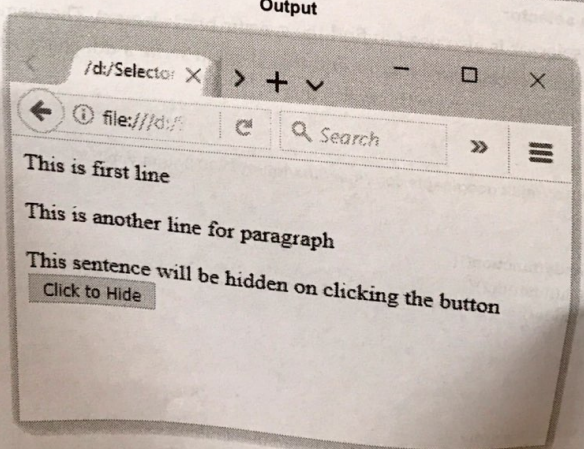
The **id** can be associated with the HTML element. In the following example we have used element to which the **id** named **myid** is associated. On button click event the function is invoked to hide the sentence written using the .

Scanned with CamScanner

To find the element with particular id write # and then name of the id.

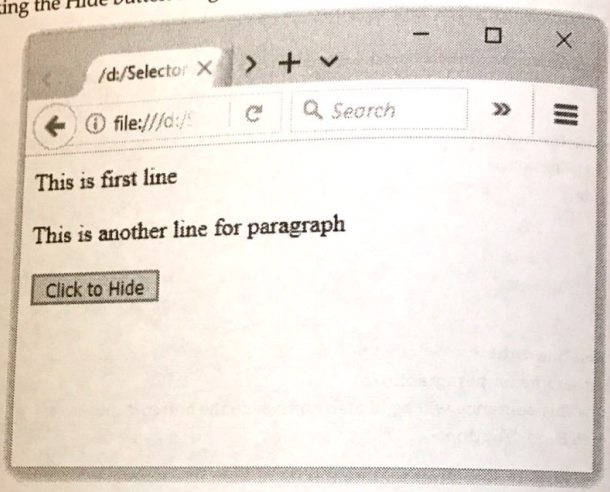
```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script>
$(document).ready(function(){
 $("button").click(function(){
  $("#myid").hide();
 });
});
</script>
</head>
<body>
<div>This is first line</div>
<p>This is another line for paragraph</p>
<div id="myid">This sentence will be hidden on clicking the button</div>
<button>Click to Hide</button>
</body>
</html>
```

Output



Scanned with CamScanner

on clicking the Hide button we get following effect



4. The .class selector

The class selector is also used to find the specific html element. The element is written along with class="*classname*" and to find this element write a dot character followed by *classname*. Following example illustrates this

```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script>
$(document).ready(function(){
 $("button").click(function(){
  $(".myclass").hide();
 });
});
</script>
</head>
<body>
<div>This is first line</div>
<p>This is another line for paragraph</p>
<div class="myclass">This sentence will be hidden on clicking the button</div>
<button>Click to Hide</button>
</body>
</html>
```

Scanned with CamScanner

Output

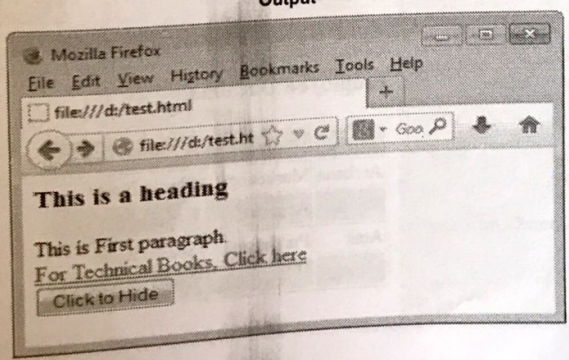
Note : The output will be the same as above

5. The href selector

The href element is used along with the anchor tag in order to introduce hyperlink in the web document. We can navigator to the desired web page using such hyperlinks. In the following program on button click, the hyperlink can be hidden.

```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script>
$(document).ready(function(){
 $("button").click(function(){
  $("[href]").hide();
 });
});
</script>
</head>

<body>
<h3>This is a heading</h3>
<div>This is First paragraph.</div>
<div>
<a href="https://www.vtubooks.com">For Technical Books, Click here</a>
</div>
<button>Click to Hide</button>
</body>
</html>
```

Output



Scanned with CamScanner

7.3.1.4 Changing Styles

To change the style the .css() function is used. The syntax is

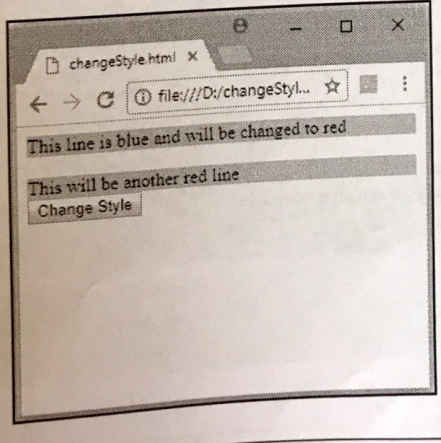
```
$(selector).css({"css property name":"css property value"});
```

For example –

JQUERY Document[changeStyle.html]

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("div").css("background-color", "red");
  });
});
</script>
</head>
<body>
<div style="background-color:blue">This line is blue and will be changed to red</div><br/>
<div style="background-color:yellow">This will be another red line</div>
<button>Change Style</button>
</body>
</html>
```

Output



Scanned with CamScanner

7.3.1.5 Creating Elements

- The jQuery after() method inserts content AFTER the selected HTML elements. The syntax is

```
$(Selector).after("Some Text");
```

- The jQuery before() method inserts content BEFORE the selected HTML elements. The syntax is

```
$(Selector).before("Some Text");
```

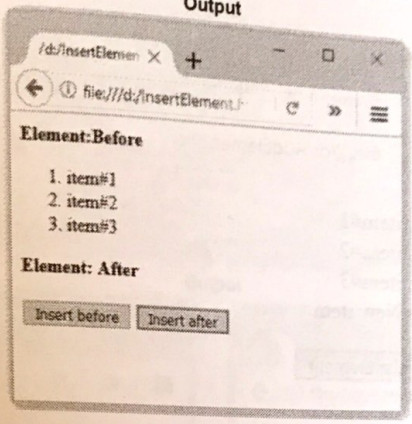
Example

JQUERY Document[InsertElement.html]

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#button1").click(function(){
    $("ol").before("<b>Element:Before</b>");
  });
  $("#button2").click(function(){
    $("ol").after("<b>Element: After</b>");
  });
});
</script>
</head>
<body>
<ol>
  <li>item#1</li>
  <li>item#2</li>
  <li>item#3</li>
</ol>
<br/><br/>
<button id="button1">Insert before</button>
<button id="button2">Insert after</button>
</body>
</html>
```

Scanned with CamScanner

Output



7.3.1.6 Appending Elements

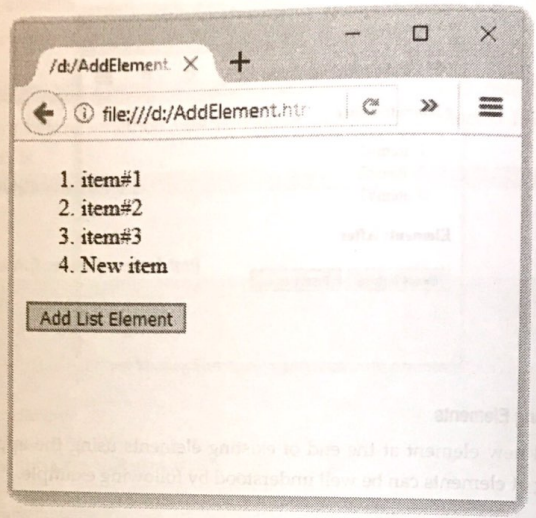
We can add new element at the end of existing elements using the append function. The appending of elements can be well understood by following example.

JQUERY Document[AddElement.html]

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#button").click(function(){
    $("ol").append("<li>New item</li>");
  });
});
</script>
</head>
<body>
<ol>
  <li>item#1</li>
  <li>item#2</li>
  <li>item#3</li>
</ol>
<button id="button">Add List Element</button>
</body>
</html>
```

Scanned with CamScanner

Output



7.3.1.7 Removing Elements

To remove elements from the DOM there are two commonly used methods – remove and empty jQuery **.empty()** method removes all the child element of the matched element where remove() method removes set of matched elements from DOM.

```
$( ".content" ).empty();
```

```
$( ".content" ).remove();
```

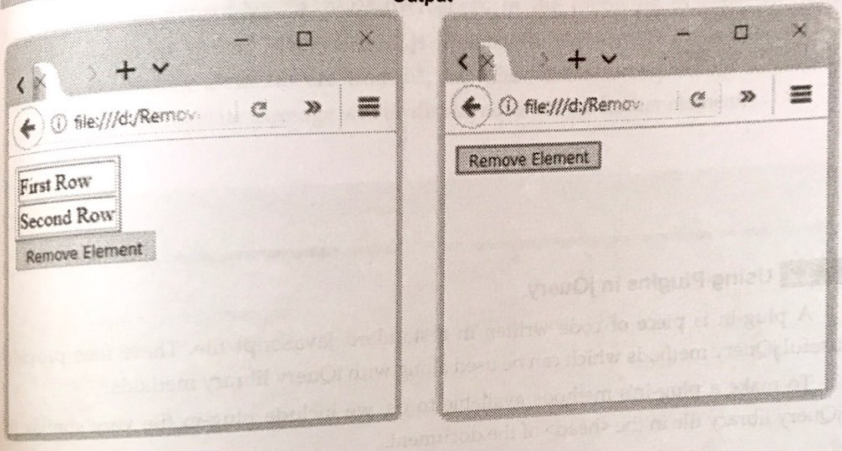
Example of .remove()

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("#tbl").remove();
  });
});
</script>
</head>
```


Scanned with CamScanner

```
<body>
<table id="tbl" border='1'>
<tr><td> First Row</td> </tr>
<tr><td> Second Row </td></tr>
</table>
<button>Remove Element</button>
</body>
</html>
```

Output



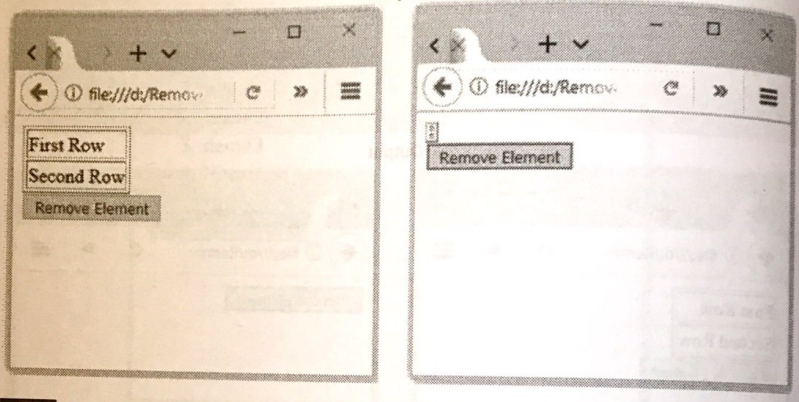
Example of .empty()

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("#tbl tr td").empty();
  });
});
</script>
</head>
<body>
 <table id="tbl" border='1'>
 <tr><td> First Row</td> </tr>
 <tr><td> Second Row </td></tr>
```


Scanned with CamScanner

```
</table>
<button>Remove Element</button>
</body>
</html>
```

Output



7.3.2 Using Plugins in jQuery

A plug-in is piece of code written in a standard JavaScript file. These files provide useful jQuery methods which can be used along with jQuery library methods.

To make a plug-in's methods available to us, we include plug-in file very similar to jQuery library file in the <head> of the document.

Creating User Defined Plugin

- We have to include a **plugin.js** file within the <script> element

```
<script src = "jquery.plug-in.js" type = "text/javascript"></script>
```

- Then we have to write a **ready** function in which the desired code for the plug-in can be written.
- The skeleton code for user defined plugin is as follows –

```
<html>
 <head>
  <title> ..... </title>
  <script type = "text/javascript"
   src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
  </script>

  <script src = "jquery.plug-in.js" type = "text/javascript"></script>
```


Scanned with CamScanner

```
  <script type = "text/javascript" language = "javascript">
    $(document).ready(function() {
      .....your custom code.....
    });
  </script>
 </head>

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

Here is an demonstration of plug-in creation. In the following example we have created a plugin file named **jquery.header.js**. This plugin will identify the header tag. As we know the header elements are from h1, h2,...,h6 , these corresponding element is identified and appropriate message will be displayed about the header element.

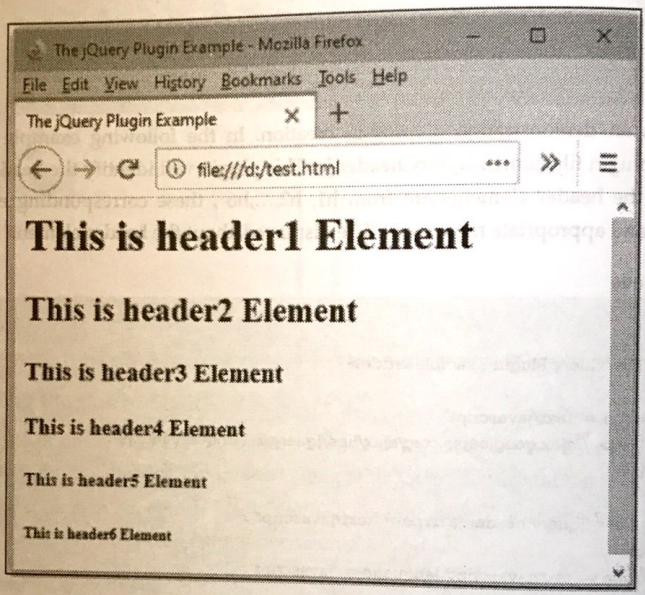
Example Code

```
<html>
 <head>
  <title>The jQuery Plugin Example</title>

  <script type = "text/javascript"
   src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
  </script>

  <script src = "jquery.header.js" type = "text/javascript">
  </script>
  <script type = "text/javascript" language = "javascript">
$(document).ready(function() {
        $("h1").warning();
        $("h2").warning();
        $("h3").warning();
        $("h4").warning();
        $("h5").warning();
        $("h6").warning();
    });
  </script>
 </head>

 <body>
  <h1>This is header1 Element</h1>
  <h2>This is header2 Element</h2>
  <h3>This is header3 Element</h3>
```

Scanned with CamScanner

```
    <h4>This is header4 Element</h4>
    <h5>This is header5 Element</h5>
    <h6>This is header6 Element</h6>
  </body>
</html>
```

Output



7.4 Creating Image Slider

- A slide show is a presentation of a series of still images on a projection screen or electronic display device, typically in a prearranged sequence.
- In JavaScript we can create a slide show by using the Array of images. Let us discuss how to create a slide show with simple JavaScript
- For creation of slide show we need to have some image files. These images can be created using some Graphics tool or some photographs.

In the following JavaScript the slide show is created and the transition from one slide to another can be possible using the **Back** and **Forward** button.

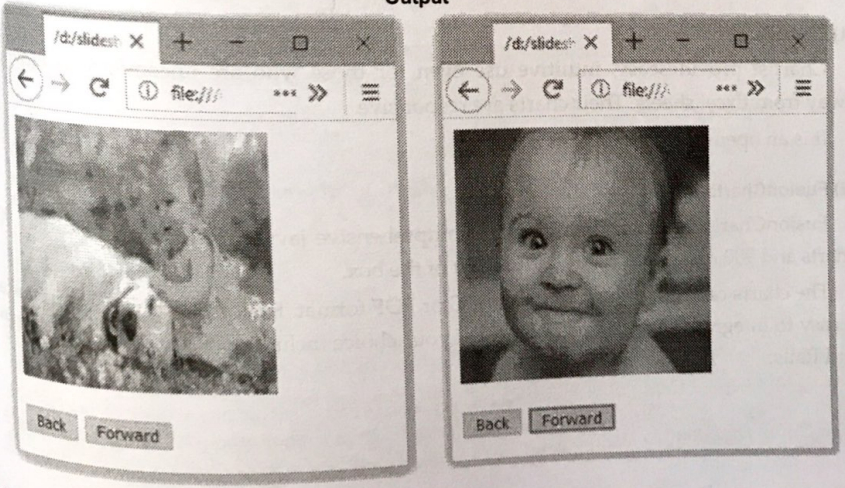
JavaScript Document

```
<!DOCTYPE html>
```


Scanned with CamScanner

```
<head>
    <script language="Javascript">
    MySlides=new Array('slide1.jpg','slide2.jpg','slide3.jpg','slide4.jpg')
    i=0
    function DisplaySlides(SlideNumber)
    {
    i=i+SlideNumber;
    if (i>MySlides.length-1)
    {
        i=0;
    }
        if (i<0)
        {
            i=MySlides.length-1;
        }
        document.SlideID.src=MySlides[i];
    }
  </script>
</head>
<body>
    <img src="slide1.jpg" name="SlideID" width="200" height="200" />
    <br/><br/>
    <input type="button" value="Back" onclick="DisplaySlides(-1)">
    <input type="button" value="Forward" onclick="DisplaySlides(1)">
</body>
</html>
```

Output



Scanned with CamScanner

Script Explanation : In above JavaScript

(1) An array named **MySlides** is created in which the four image files act as four slides.

(2) In <body> section, the image tag is used to place the first slide.

(3) Then two buttons are placed – one for moving back in slide show and anther is for moving forward in slide show. For each transition(either forward or back the function **DisplaySlides** will be called by passing initial –1 or +1 value.

(4) **DisplaySlides** is a simple function in which the index i of slide is incremented by one. When the slide show reaches to maximum slide number it is reset and the slide show is possible from beginning.

Review Question

1. *Explain how to generate a slide show using any suitable scripting language.*

7.5 Generating Charts from Data using Third Party Libs

- Every business is heavily dependent upon the data analysis for important decisions.
- JavaScript charting libraries have emerged as the most powerful tools for visualizing data in the form of beautiful, easy to understand, interactive charts.
- They make it easier to extract and convey key patterns and insights that are often not apparent with static charts.
- Following are some commonly used JavaScript charting libraries.

(1) Chartist.js

Chartist provideseasy, intuitive use even for those who are uncomfortable moving away from Excel sheets. Their charts are responsive.

It is an opensource library for charting.

(2) FusionCharts.js

FusionCharts brings in one of most comprehensive javascript libraries with over 90 charts and 900 maps – all ready for use out of the box.

The charts can be exported in PNG, JPG or PDF format. FusionCharts extensions make it easy to integrate with any technology of your choice including jQuery, AngularJS, PHP and Rails.

Scanned with CamScanner

(3) Chart.js

Chart.js is perfect for small projects – when you need flat, clean, elegant javascript charts, fast. It is a tiny open source library at just 11 kb when minified and zipped. This includes 6 core chart types (line, bar, radar, polar, pie and doughnut), each in its own module, so you can even load

(4) Google Charts

Google charts provides a wide range of charts, for almost any kind of data visualization need. The charts are based on HTML5/SVG and VML for older IE versions. All charts are interactive, and some are pannable/zoomable as well.

This charting software is available as an open source product.

Let us now discuss how to create chart using Google chart

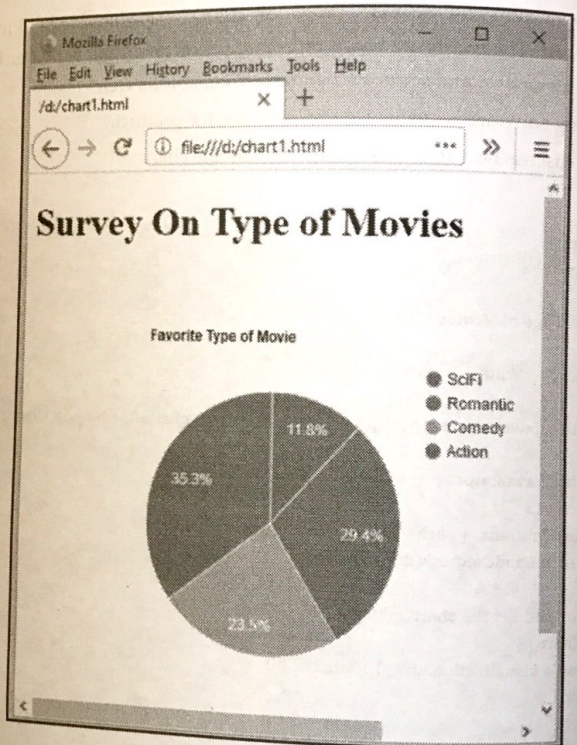
Example Code

```
<!DOCTYPE html>
<html lang="en-US">
<body>
<h1>Survey On Type of Movies</h1>

<div id="piechart"></div>

<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>

<script type="text/javascript">
// Load google charts
google.charts.load('current', {'packages':['corechart']});
google.charts.setOnLoadCallback(drawChart);

// Draw the chart and set the chart values
function drawChart() {
 var data = google.visualization.arrayToDataTable([
 ['Type', 'Opinion'],
 ['SciFi', 2],
 ['Romantic', 5],
 ['Comedy', 4],
 ['Action',6]
]);

 // Optional; add a title and set the width and height of the chart
 var options = {'title':'Favorite Type of Movie', 'width':550, 'height':400};
```

Scanned with CamScanner

```
// Display the chart inside the <div> element with id="piechart"
var chart = new google.visualization.PieChart(document.getElementById('piechart'));
chart.draw(data, options);
}
</script>
</body>
</html>
```

Output



Explanation :

(1) To add the reference to the chart API at google.com, we must include following line in the beginning –

```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
```

(2) This line loads the loader itself. You can only load the loader one time no matter how many charts you plan to draw. After loading the loader, you can call the

Scanned with CamScanner

google.charts.load function one or more times to load packages for particular chart types.

(3) The first argument to **google.charts.load** is the version name or number, as a string. If you specify '**current**', this causes the latest official release of Google Charts to be loaded. If you want to try the candidate for the next release, use '**upcoming**' instead.

(4) Another argument to google.charts.load is **corechart.** The example above assumes you want to display a corechart (bar, column, line, area, stepped area, bubble, pie, donut, combo, candlestick, histogram, scatter).

(5) If you want to draw more than one chart, you can either register more than one callback function using **setOnLoadCallback**, or you can combine them into one function

(6) Now we use function **drawChart()**. Inside this function we prepare data table. All charts require data. Google Chart Tools charts require data to be wrapped in a JavaScript class called **google.visualization.DataTable.** A **DataTable** is a two-dimensional table with rows and columns. In above example we have created the data table as

```
var data = google.visualization.arrayToDataTable([
['Type', 'Opinion'],
['SciFi', 2],
['Romantic', 5],
['Comedy', 4],
['Action',6]
]);
```

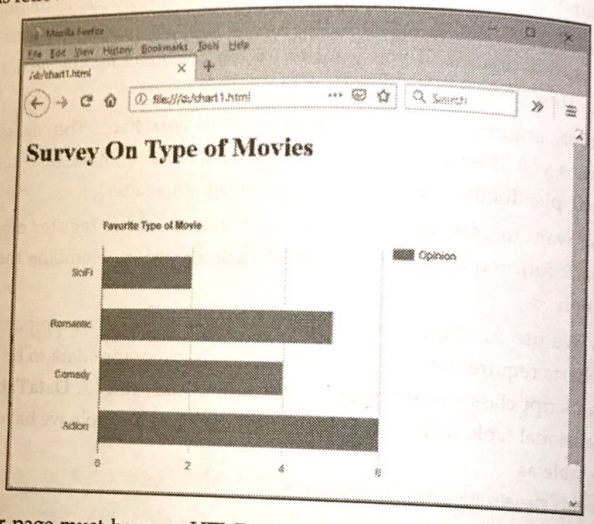
(7) Every chart has many customizable options, including title, colors, line thickness, background fill, and so on. We have used –

```
var options = {'title':'Favorite Type of Movie', 'width':550, 'height':400};
```

(8) Each chart type is based on a different class. For instance, the pie chart is based on the **google.visualization.PieChart** class, the bar chart is based on the **google.visualization.BarChart** class, and so on. Both pie and bar charts are included in the **corechart** package.

Scanned with CamScanner

In above example, if you change the **google.visualization.PieChart** to **google.visualization.BarChart** then the data is displayed in the form of bar chart. It would be as follow –



Your page must have an HTML element (typically a) to hold your chart. You must pass your chart a reference to this element, so assign it an ID that you can use to retrieve a reference using **document.getElementById()**. Anything inside this element will be replaced by the chart when it is drawn.

(9) Every chart supports a **draw()** method that takes two values: a DataTable (or a DataView) object that holds your data, and an optional chart options object. The options object is not required, and you can ignore it or pass in null to use the chart's default options.

After you call **draw()**, your chart will be drawn on the page. You should call the **draw()** method every time you change the data or the options and want to update the chart.

Review Questions

1. *Enlist and give brief introduction of third part libraries used for generating charts.*
2. *Explain the in detail how to generate a chart using any third part library*

Scanned with CamScanner