WEB PROGRAMMING with PHP

BCA Semester – 2

Unit 1 – PHP Basics

Static and Dynamic Website

- **Static Website (Web Content):** Static website is the basic type of website that is easy to create. You don't need the knowledge of web programming and database design to create a static website. Its web pages are coded in HTML. The codes are fixed for each page so the information contained in the page does not change and it looks like a printed page.

- **Dynamic website (Web Content):** Dynamic website is a collection of dynamic web pages whose content changes dynamically. It accesses content from a database or Content Management System (CMS). Therefore, when you alter or update the content of the database, the content of the website is also altered or updated. Dynamic website uses client-side scripting or server-side scripting, or both to generate dynamic content. Client-side scripting generates content at the client computer on the basis of user input. The web browser downloads the web page from the server and processes the code within the page to render information to the user. In server-side scripting, the software runs on the server and processing is completed in the server then plain pages are sent to the user.

Static Website	Dynamic Website
Prebuilt content is same every time the page is loaded.	Content is generated quickly and changes regularly.
It uses the **HTML** code for developing a website.	It uses the server-side languages such as **PHP, SERVLET, JSP, and ASP.NET** etc. for developing a website.
It sends exactly the same response for every request.	It may generate different HTML for each of the request.
The content is only changed when someone publishes and updates the file (sends it to the web server).	The page contains "server-side" code which allows the server to generate the unique content when the page is loaded.
Flexibility is the main advantage of static website.	Content Management System (CMS) is the main advantage of dynamic website.

Client-side and Server-Side Scripting

- **Client-side scripting:** Web browsers execute client-side scripting. It is used when browsers have all code. Source code is used to transfer from webserver to user’s computer over the internet and run directly on browsers. It is also used for validations and functionality for user events. It allows for more interactivity. It usually performs several actions without going to the user. It cannot be basically used to connect to databases on a web server. These scripts cannot access the file system that resides in the web browser. Pages are altered on basis of the user’s choice. It can also be used to create “cookies” that store data on the user’s computer.

- **Server-side scripting:** Web servers are used to execute server-side scripting. They are basically used to create dynamic pages. It can also access the file system residing at the webserver. A server-side environment that runs on a scripting language is a web server. Scripts can be written in any of a number of server-side scripting languages available. It is used to retrieve and generate content for dynamic pages. It is used to require to download plugins. In this load times are generally faster than client-side scripting. When you need to store and retrieve information a database will be used to contain data. It can use huge resources of the server. It reduces client-side computation overhead. The server sends pages to the request of the user/client.

BASIS FOR COMPARISON	SERVER-SIDE SCRIPTING	CLIENT-SIDE SCRIPTING
Basic	Works in the back end which could not be visible at the client end.	Works at the front end and script are visible among the users.
Processing	Requires server interaction.	Does not need interaction with the server.
Languages involved	PHP, ASP.net, Ruby on Rails, ColdFusion, Python, etcetera.	HTML, CSS, JavaScript, etc.
Affect	Could effectively customize the web pages and provide dynamic websites.	Can reduce the load to the server.
Security	Relatively secure.	Insecure

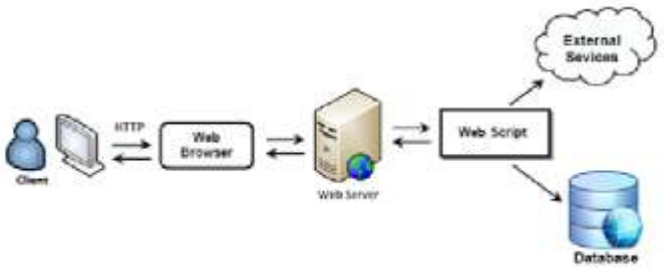
Introduction to other server-side languages

After the advent of CGI, multiple programming languages were evolved such as PHP, Python, Ruby, ColdFusion, C#, Java, C++ and so on for server-side scripting among which some of them are described below:

- **PHP:** It is the most prevalent server-side language used on the web which was designed to extract and manipulate information in the database. The language is used in association with SQL language for the Database. It is used in Facebook, WordPress and Wikipedia.
- **Python:** The language is fast and contains shorter code. It is good for beginners as it concentrates on the readability and simplicity of the code. Python functions well in the object-oriented environment and used in famous sites like Youtube, Google, etc.
- **Ruby:** It contains complex logic which packages the back-end with database utility which can also be provided by PHP and SQL.

Webserver

The term web server can refer to hardware or software, or both of them working together. On the hardware side, a web server is a computer that stores web server software and a website's component files (for example, HTML documents, images, CSS stylesheets, and JavaScript files). A web server connects to the Internet and supports physical data interchange with other devices connected to the web. On the software side, a web server includes several parts that control how web users access hosted files. At a minimum, this is an HTTP server. An HTTP server is software that understands URLs (web addresses) and HTTP (the protocol your browser uses to view webpages). An HTTP server can be accessed through the domain names of the websites it stores, and it delivers the content of these hosted websites to the end user's device. At the most basic level, whenever a browser needs a file that is hosted on a web server, the browser requests the file via HTTP. When the request reaches the correct (hardware) web server, the (software) HTTP server accepts the request, finds the requested document, and sends it back to the browser, also through HTTP. (If the server doesn't find the requested document, it returns a 404 response instead.)



Web server software is accessed through the domain names of websites and ensures the delivery of the site's content to the requesting user. The software side is also comprised of several components, with at least an HTTP server. The HTTP server is able to understand HTTP and URLs. As hardware, a web server is a computer that stores web server software and other files related to a website, such as HTML documents, images and JavaScript files. When a web browser, like Google Chrome or Firefox, needs a file that's hosted on a web server, the browser will request the file by HTTP. When the request is received by the web server, the HTTP server will accept the request, find the content and send it back to the browser through HTTP. Many basic web servers will also support server-side scripting, which is used to employ scripts on a web server that can customize the response to the client. Server-side scripting runs on the server machine and typically has a broad feature set, which includes database access. The server-side scripting process will also use Active Server Pages (ASP), Hypertext Pre-processor (PHP) and other scripting languages. This process also allows HTML documents to be created dynamically.

Dynamic vs. static web servers: A web server can be used to serve either static or dynamic content. Static refers to the content being shown as is, while dynamic content can be updated and changed. A **static web** server will consist of a computer and HTTP software. It is considered static because the sever will send hosted files as is to a browser. **Dynamic web** browsers will consist of a web server and other software such as an application server and database. It is considered dynamic because the application server can be used to update any hosted files before they are sent to a browser. The web server can generate content when it is requested from the database. Though this process is more flexible, it is also more complicated.

Web server software: There are a number of common web servers available, some including:

- **Apache HTTP Server.** Developed by Apache Software Foundation, it is a free and open source web server for Windows, Mac OS X, Unix, Linux, Solaris and other operating systems; it needs the Apache license.
- **Microsoft Internet Information Services (IIS)**. Developed by Microsoft for Microsoft platforms; it is not open sourced, but widely used.
- **Nginx**. A popular open source web server for administrators because of its light resource utilization and scalability. It can handle many concurrent sessions due to its event-driven architecture. Nginx also can be used as a proxy server and load balancer.
- **Lighttpd**. A free web server that comes with the FreeBSD operating system. It is seen as fast and secure, while consuming less CPU power.
- **Sun Java System Web Server**. A free web server from Sun Microsystems that can run on Windows, Linux and Unix. It is well-equipped to handle medium to large websites.

HTTP, HTTPS, and FTP protocol

HTTP: The Hypertext Transfer Protocol (HTTP) is application-level protocol for collaborative, distributed, hypermedia information systems. It is the data communication protocol used to establish communication between client and server. HTTP is TCP/IP based communication protocol, which is used to deliver the data like image files, query results, HTML files etc on the World Wide Web (WWW) with the default port is TCP 80. It provides the standardized way for computers to communicate with each other. The Basic Characteristics of HTTP (Hyper Text Transfer Protocol): It is the protocol that allows web servers and browsers to exchange data over the web. It is a request response protocol. It uses the reliable TCP connections by default on TCP port 80. It is stateless means each request is considered as the new request. In other words, server doesn't recognize the user by default. There are three fundamental features that make the HTTP a simple and powerful protocol used for communication:

- **HTTP is media independent:** It specifies that any type of media content can be sent by HTTP as long as both the server and the client can handle the data content.
- **HTTP is connectionless:** It is a connectionless approach in which HTTP client i.e., a browser initiates the HTTP request and after the request is sent the client disconnects from server and waits for the response.
- **HTTP is stateless:** The client and server are aware of each other during a current request only. Afterwards, both of them forget each other. Due to the stateless nature of protocol, neither the client nor the server can retain the information about different request across the web pages.

HTTPS: HTTPS is an abbreviation of Hypertext Transfer Protocol Secure. It is a secure extension or version of HTTP. This protocol is mainly used for providing security to the data sent between a website and the web browser. It is widely used on the internet and used for secure communications. This protocol uses the 443 port number for communicating the data. This protocol is also called HTTP over SSL because the HTTPS communication protocols are encrypted using the SSL (Secure Socket Layer). By default, it is supported by various web browsers. Those websites which need login credentials should use the HTTPS protocol for sending the data.

HTTP	HTTPS
It is an abbreviation of Hypertext Transfer Protocol	It is an abbreviation of Hypertext Transfer Protocol Secure.
This protocol operates at the application layer.	This protocol operates at the transport layer.
The data which is transferred in HTTP is plain text.	The data which is transferred in HTTPS is encrypted, i.e., ciphertext.
By default, this protocol operates on port number 80.	By default, this protocol operates on port number 443.
The URL (Uniform Resource Locator) of HTTP start with http://	The URL (Uniform Resource Locator) of HTTPS start with https://
This protocol does not need any certificate.	But, this protocol requires an SSL (Secure Socket Layer) certificate.
Encryption technique is absent in HTTP.	Encryption technique is available or present in HTTPS.
The speed of HTTP is fast as compared to HTTPS.	The speed of HTTPS is slow as compared to HTTP.
It is un-secure.	It is highly secure.
Examples of HTTP websites are Educational Sites, Internet Forums, etc.	Examples of HTTPS websites are shopping websites, banking websites, etc.

Introduction to PHP

The term PHP is an acronym for **Hypertext Preprocessor**. It is a server-side scripting language that is used for web development. It can be easily embedded with HTML files. HTML codes can also be written in a PHP file. The PHP codes are executed on the server-side whereas HTML codes are directly executed on the browser. PHP is an open-source, interpreted, and object-oriented scripting language that can be executed at the server-side. PHP is well suited for web development. Therefore, it is used to develop web applications (an application that executes on the server and generates the dynamic page.). PHP was created by **Rasmus Lerdorf** in 1994 but appeared in the market in 1995. PHP 7.4.0 is the latest version of PHP, which was released on 28 November. Some important points need to be noticed about PHP are as followed:

- **Easy to Learn:** It is easier to learn for anyone who has come across to any programming language for the first time.
- **Free of Cost:** Since it is an open-source language, therefore developers are allowed to use its components and all methods for free.
- Flexible: Since It is a dynamically typed language, therefore there are no hard rules on how to build features using it.
- **Supports nearly all databases:** It supports all the widely used databases, including MySQL, ODBC, SQLite etc.
- **Secured:** It has multiple security levels provides us a secure platform for developing websites as it has multiple security levels.
- **Huge Community Support:** It is loved and used by a huge number of developers. The developers share their knowledge with other people of the community that want to know about it.

Applications of PHP: As mentioned before, PHP is one of the most widely used language over the web. I'm going to list few of them here:

- PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them.

- PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user.
- You add, delete, modify elements within your database through PHP.
- Access cookies variables and set cookies.
- Using PHP, you can restrict users to access some pages of your website.
- It can encrypt data.

Advantages of PHP:

- It is supported by all Operating Systems like Windows, Linux, Unix, etc.
- It is integrated with other programming languages (HTML, CSS, JavaScript, etc) and databased.
- It is easy to connect with the database to store and retrieve data from the database. Multiple databases can also be integrated with PHP.
- It is the fastest programming language compared to other programming languages.
- PHP frameworks and tools are used to protect web applications from outer attacks and security threats.

Disadvantages of PHP:

- Since PHP is open-source so its code is visible to all programmers. If any bugs exist in the source code then its weakness can be explored by other programmers.
- It is not suitable for large applications because its maintenance is difficult.
- The error handling of a PHP framework is not good.

PHP configuration in IIS & Apache Web server

In order to develop and run PHP Web pages three vital components need to be installed on your computer system.

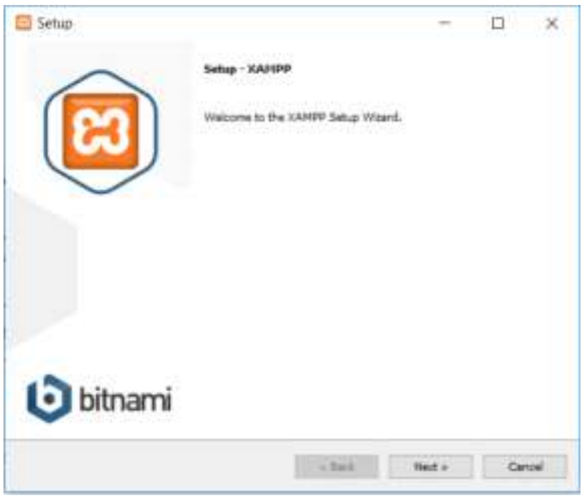
- **Web Server** − PHP will work with virtually all Web Server software, including Microsoft's Internet Information Server (IIS) but then most often used is freely available Apache Server. Download Apache for free here − https://httpd.apache.org/download.cgi
- **Database** − PHP will work with virtually all database software, including Oracle and Sybase but most commonly used is freely available MySQL database. Download MySQL for free here − https://www.mysql.com/downloads/
- **PHP Parser** − In order to process PHP script instructions a parser must be installed to generate HTML output that can be sent to the Web Browser. This tutorial will guide you how to install PHP parser on your computer.

Using all in one package we can use following Web Server Software:

- WAMPP – Windows Apache MySQL PHP Perl
- LAMPP – Linux Apache MySQL PHP Perl
- MAMPP – Macintosh Apache MySQL PHP
- XAMPP – X-OS (Cross platform) Apache MySQL PHP Perl (Recommended)

Here are easy and almost error-free method to install PHP on a local machine which is by using all in one package called XAMPP.

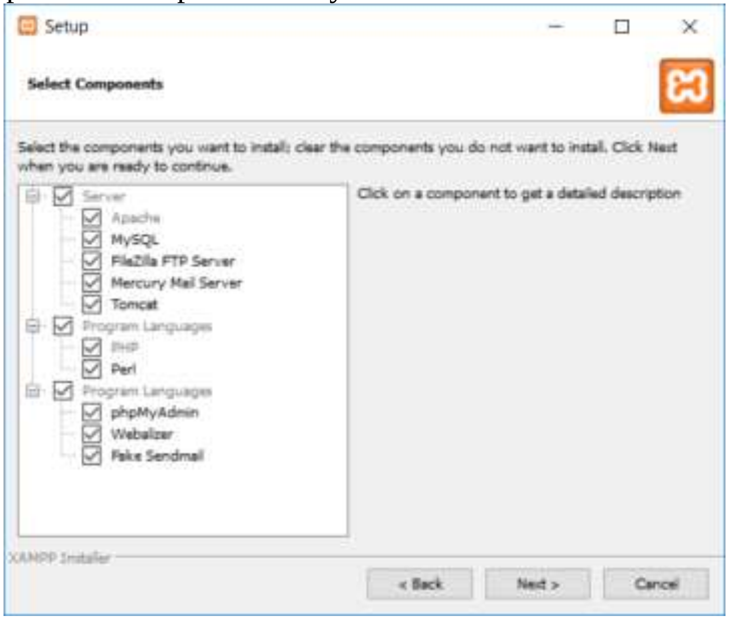
You can download the XAMPP software from the official website here with the latest windows version and latest PHP version. Open the downloaded .exe file: After opening the downloaded file you will see a popup from windows, click yes and proceed further. Click on ‘next’: You’ll see a welcome window of XAMPP like below, click next.


Setup

Setup - XAMPP

Welcome to the XAMPP Setup Wizard.

Select the components: Select the components you want to install. Please select the MySQL and phpMyAdmin components, all other components are optional that you can unchecked.


Setup

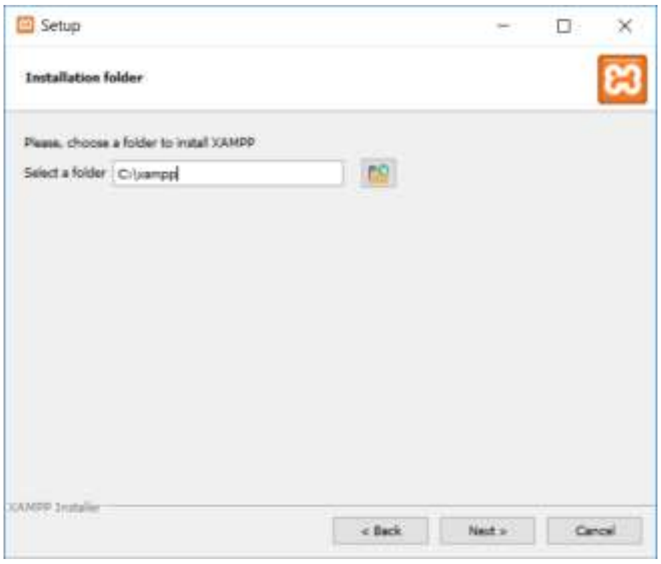
Select Components

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- [✓] Server
  - [✓] Apache
  - [✓] MySQL
  - [✓] FileZilla FTP Server
  - [✓] Mercury Mail Server
  - [✓] Tomcat
- [✓] Program Languages
  - [✓] PHP
  - [✓] Perl
- [✓] Program Languages
  - [✓] phpMyAdmin
  - [✓] Webalizer
  - [✓] Fake Sendmail

Click on a component to get a detailed description

Select the installation location: Select the location you want to install the XAMPP, the default is C:\xampp.

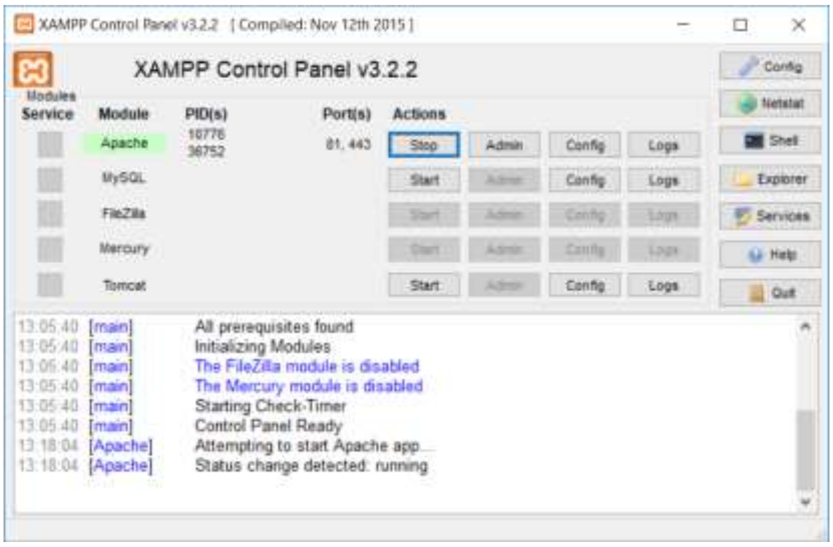

Setup

Installation folder

Please, choose a folder to install XAMPP

Select a folder C:\xampp

Click next: Your clicking next, your installation will begin. Open XAMPP control panel: After successfully installing the XAMPP in your local machine open the control panel by searching in the windows search bar for ‘XAMPP control panel’ or by going to the installation directory of XAMPP. You’ll see a window like below.


XAMPP Control Panel v3.2.2 [Compiled: Nov 12th 2015]
XAMPP Control Panel v3.2.2
Config
Netstat
Shell
Explorer
Services
Help
Quit

```
Modules
Service  Module    PID(s)  Port(s)  Actions
         Apache    10776   81, 443  Stop   Admin  Config  Logs
                   36752
         MySQL                      Start  Admin  Config  Logs
         FileZilla                  Start  Admin  Config  Logs
         Mercury                    Start  Admin  Config  Logs
         Tomcat                     Start  Admin  Config  Logs

13:05:40  [main]    All prerequisites found
13:05:40  [main]    Initializing Modules
13:05:40  [main]    The FileZilla module is disabled
13:05:40  [main]    The Mercury module is disabled
13:05:40  [main]    Starting Check-Timer
13:05:40  [main]    Control Panel Ready
13:18:04  [Apache]  Attempting to start Apache app...
13:18:04  [Apache]  Status change detected: running
```

Checking PHP installation: Create a php file in **htdocs** folder in your installation directory. (C:/XAMPP/htdocs) and add the following code in it.

Understanding of PHP.INI file

At the time of PHP installation, php.ini is a special file provided as a default configuration file. It's very essential configuration file which controls, what a user can or cannot do with the website. Each time PHP is initialized, the php.ini file is read by the system. Sometimes you need to change the behavior of PHP at runtime, then this configuration file is to use. All the settings related to register global variables, upload maximum size, display log errors, resource limits, the maximum time to execute a PHP script and others are written in a file as a set of directives which helps in declaring changes. Note: Whenever some changes are performed in the file, you need to restart our web server. It helps in easy administration of web server using these configuration files. We can also write our own custom configuration files. To check file path use the following program:

```
<?php
	echo phpinfo();
?>
```

Environment variables of php.ini file:

memory_limit: This setting is done to show the maximum amount of memory a script consumes.

Important settings or common parameters of the php.ini file:

- enable_safe_mode = on Its default setting to ON whenever PHP is compiled. Safe mode is most relevant to CGI use.
- register_globals = on its default setting to ON which tells that the contents of EGPCS (Environment, GET, POST, Cookie, Server) variables are registered as global variables. But due to a security risk, the user has to ensure if it set to OFF for all scripts.
- upload_max_filesize This setting is for the maximum allowed size for uploaded files in the scripts.
- upload_tmp_dir = [DIR] Don't uncomment this setting.
- post_max_size This setting is for the maximum allowed size of POST data that PHP will accept.
- display_errors = off This setting will not allow showing errors while running PHP project in the specified host.
- error_reporting = E_ALL & ~E_NOTICE: This setting has default values as E_ALL and ~E_NOTICE which shows all errors except notices.
- error_prepend_string = [""] This setting allow you to make different color of messages.
- max_execution_time = 30 Maximum execution time is set to seconds for any script to limit the time in production servers.
- short_open_tags = Off To use XML functions, we have to set this option as off.
- session.save-handler = files You don't need to change anything in this setting.
- variables_order = EGPCS This setting is done to set the order of variables as Environment, GET, POST, COOKIE, SERVER. The developer can change the order as per the need also.
- warn_plus_overloading = Off This setting issues a warning if + used with strings in a form of value.
- gpc_order = GPC This setting has been GPC Deprecated.
- magic_quotes_gpc = on This setting is done in case of many forms are used which submits to themselves or others and display form values.

- magic_quotes_runtime = Off If magic_quotes_sybase is set to On, this must be Off, this setting escape quotes.
- magic_quotes_sybase = Off If this setting is set to off it should be off, this setting escape quotes.
- auto-prepend-file = [filepath] This setting is done when we need to automatically include() it at the beginning of every PHP file.
- auto-append-file = [filepath] This setting is done when we need to automatically include() it at the end of every PHP file.
- include_path = [DIR] This setting is done when we need to require files from the specified directories. Multiple directories are set using colons.
- ignore_user_abort = [On/Off] This settings control what will happen when the user click any stop button. The default value is on this setting doesn't work on CGI mode it works on only module mode.
- doc_root = [DIR] This setting is done if we want to apply PHP to a portion of our website.
- file_uploads = [on/off] This flag is set to ON if file uploads are included in the PHP code.
- mysql.default_host = hostname This setting is done to connect to MySQL default server if no other server host is mentioned.
- mysql.default_user = username This setting is done to connect MySQL default username, if no other name is mentioned.
- mysql.default_password = password This setting is done to connect MySQL default password if no other password is mentioned.

Understanding of PHP .htaccess file

The .htaccess (Hypertext Access) file is an Apache distributed server configuration file. You can use the .htaccess file to set server configurations for a specific directory. This directory can be the root directory of your website or another subdirectory where the .htaccess file is created in order to enable extra features for that subdirectory. You can use the .htaccess file to modify various configurations and thus make changes to your website. These changes include authorization, error handling, redirects for specific URLs, user permissions, etc. Like any other Apache configuration file, the .htaccess file is read from top to bottom. That is, the above configurations are performed before those below. Common uses of the .htaccess file:

- **Change the default start page:** Suppose you want to change your home page (e.g. index.html) with some other HTML page (e.g. home.html) while keeping the index.html file intact, you can change the default landing page by adding the code below in your .htaccess file. In the configuration file, it is also possible to add more than one file. Here in this example, first, the server will check for index.html, if it does not find a file with that name, it continues to home.htm and so on.

- **Block a specific IP or range of IPs:** You can also block a specific IP address or a range of IP addresses from visiting your website. To do this, you need to add these lines to your .htaccess file: To restrict access from certain countries, you must obtain the IP address ranges assigned to that particular country. It is important to note that this method is not 100% efficient as the IP address assignments may change and the IP address ranges may overlap. Even so, this method blocks most of the traffic from the specified countries.
- **301 Permanent Redirect:** 301 is an HTTP response code to your web browser from the webserver. A 301 status code indicates that the requested resources have been permanently moved to a new URL. 301 redirects are very useful when a page is no longer relevant or the page is deleted. You can use the below code to apply 301 redirects.
- **WWW to non-WWW and non-WWW to WWW:** As search engines consider "www" and "non-www" URLs two different things, so redirecting requests from non-preferred domains becomes very important. Let's take an example of "www.example.com"
- **Redirect from HTTP to HTTPS:** There are two main reasons, one is Security because it ensures that the user data is encrypted from the user browser to the webserver and the second reason is the SEO(Search Engine Optimization) because HTTPS websites have higher advantages of ranking over HTTP websites. If you want to transfer the entire traffic of your website from HTTP to HTTPS, that you will need to add the following to your .htaccess file.
- **Customize your Error Page:** If you want to customize your 404 error page, you can define your own custom error page in the .htaccess file. Just copy the below text in your .htaccess file.
- **Authenticated folder:** For authentication purposes, you can protect a directory of an application by adding the code given below in your .htaccess file. Once the .htaccess file is updated your directory will require a username and password to access it.

PHP Variable

In PHP, a variable is declared using a $ sign followed by the variable name. Here, some important points to know about variables: As PHP is a loosely typed language, so we do not need to declare the data types of the variables. It automatically analyzes the values and makes conversions to its correct datatype. After declaring a variable, it can be reused throughout the code. Assignment Operator (=) is used to assign the value to a variable. Syntax of declaring a variable in PHP is given below:

```
$variablename=value;
```

Rules for declaring PHP variable:

- A variable must start with a dollar ($) sign, followed by the variable name.
- It can only contain alpha-numeric character and underscore (A-z, 0-9, _).
- A variable name must start with a letter or underscore (_) character.
- A PHP variable name cannot contain spaces.
- One thing to be kept in mind that the variable name cannot start with a number or special symbols.
- PHP variables are case-sensitive, so $name and $NAME both are treated as different variable.

PHP Variable Scope

The scope of a variable is defined as its range in the program under which it can be accessed. In other words, "The scope of a variable is the portion of the program within which it is defined and can be accessed." PHP has three types of variable scopes: Local variable, Global variable, and Static variable.

Local variable: The variables that are declared within a function are called local variables for that function. These local variables have their scope only in that particular function in which they are declared. This means that these variables cannot be accessed outside the function, as they have local scope. A variable declaration outside the function with the same name is completely different from the variable declared inside the function. Let's understand the local variables with the help of an example:

```
<?php
    function local_var()
    {
        $num = 45;  //local variable
        echo "Local variable declared inside the function is: ". $num;
    }
    local_var();
?>
```

Output:

```
Local variable declared inside the function is: 45
```

Global variable: The global variables are the variables that are declared outside the function. These variables can be accessed anywhere in the program. To access the global variable within a function, use the GLOBAL keyword before the variable. However, these variables can be directly accessed or used outside the function without any keyword. Therefore, there is no need to use any keyword to access a global variable outside the function. Let's understand the global variables with the help of an example:

```
<?php
    $name = "Geetanjali";  //Global Variable
    function global_var()
    {
        global $name;
        echo "Variable inside the function: ". $name;
        echo "</br>";
    }
    global_var();
    echo "Variable outside the function: ". $name;
?>
```

Output:

```
Variable inside the function: Geetanjali
Variable outside the function: Geetanjali
```

Static variable: It is a feature of PHP to delete the variable, once it completes its execution and memory is freed. Sometimes we need to store a variable even after completion of function execution. Therefore, another important feature of variable scoping is static variable. We use the static keyword before the variable to define a variable, and this variable is called as static variable. Static variables exist only in a local function, but it does not free its memory after the program execution leaves the scope. Understand it with the help of an example:

```
<?php
    function static_var()
    {
        static $num1 = 3;       //static variable
        $num2 = 6;          //Non-static variable
        //increment in non-static variable
        $num1++;
        //increment in static variable
        $num2++;
        echo "Static: " .$num1 ."</br>";
        echo "Non-static: " .$num2 ."</br>";
    }

//first function call
    static_var();

    //second function call
    static_var();
?>
```

Output:

```
Static: 4
Non-static: 7
Static: 5
Non-static: 7
```

GET & POST method

There are two ways the browser client can send information to the web server. The GET Method and The POST Method. Before the browser sends the information, it encodes it using a scheme called URL encoding. In this scheme, name/value pairs are joined with equal signs and different pairs are separated by the ampersand.

```
name1=value1&name2=value2&name3=value3
```

Spaces are removed and replaced with the + character and any other nonalphanumeric characters are replaced with a hexadecimal value. After the information is encoded it is sent to the server.

The GET Method: The GET method sends the encoded user information appended to the page request. The page and the encoded information are separated by the ? character.

```
http://www.test.com/index.htm?name1=value1&name2=value2
```

The GET method produces a long string that appears in your server logs, in the browser's Location: box. The GET method is restricted to send upto 1024 characters only. Never use GET method if you have password or other sensitive information to be sent to the server. GET can't be used to send binary data, like images or word documents, to the server. The data sent by GET method can be accessed using QUERY_STRING environment variable. The PHP provides $_GET associative array to access all the sent information using GET method. Try out following example by putting the source code in test.php script.

```
<?php
  if( $_GET["name"] || $_GET["age"] ) {
    echo "Welcome ". $_GET['name']. "<br />";
    echo "You are ". $_GET['age']. " years old.";
    exit();
  }
?>
<html>
  <body>

    <form action = "<?php $_PHP_SELF ?>" method = "GET">
      Name: <input type = "text" name = "name" />
      Age: <input type = "text" name = "age" />
      <input type = "submit" />
    </form>

  </body>
</html>

```

The POST Method: The POST method transfers information via HTTP headers. The information is encoded as described in case of GET method and put into a header called QUERY_STRING. The POST method does not have any restriction on data size to be sent. The POST method can be used to send ASCII as well as binary data. The data sent by POST method goes through HTTP header so security depends on HTTP protocol. By using Secure HTTP you can make sure that your information is secure. The PHP provides $_POST associative array to access all the sent information using POST method. Try out following example by putting the source code in test.php script.

```
<?php
  if( $_POST["name"] || $_POST["age"] ) {
    if (preg_match("/[^A-Za-z'-]/",$_POST['name'] )) {
      die ("invalid name and name should be alpha");
    }
    echo "Welcome ". $_POST['name']. "<br />";
    echo "You are ". $_POST['age']. " years old.";

    exit();
  }
?>
<html>
  <body>
    <form action = "<?php $_PHP_SELF ?>" method = "POST">
      Name: <input type = "text" name = "name" />
      Age: <input type = "text" name = "age" />
      <input type = "submit" />
    </form>
  </body>
</html>


```

The $_REQUEST variable: The PHP $_REQUEST variable contains the contents of both $_GET, $_POST, and $_COOKIE. We will discuss $_COOKIE variable when we will explain about cookies. The PHP $_REQUEST variable can be used to get the result from form data sent with both the GET and POST methods. Try out following example by putting the source code in test.php script.

```
<?php
  if( $_REQUEST["name"] || $_REQUEST["age"] ) {
    echo "Welcome ". $_REQUEST['name']. "<br />";
    echo "You are ". $_REQUEST['age']. " years old.";
    exit();
  }
?>
<html>
```

```
  <body>

    <form action = "<?php $_PHP_SELF ?>" method = "POST">
      Name: <input type = "text" name = "name" />
      Age: <input type = "text" name = "age" />
      <input type = "submit" />
    </form>

  </body>
</html>
```

Here $_PHP_SELF variable contains the name of self script in which it is being called.

PHP Operator

What is Operator? Simple answer can be given using expression 4 + 5 is equal to 9. Here 4 and 5 are called operands and + is called operator. PHP language supports following type of operators.

- Arithmetic Operators (+, -, *, /, %)
- Unary Operators (++, --)
- Comparison Operators (==, !=, <, >, <=, >=)
- Logical Operators (&&, and, ||, or, !)
- Assignment Operators (=, +=, -=, *=, *=, %=)
- Conditional (or ternary) Operators (? :)

Conditional Structure

The if, elseif ...else and switch statements are used to take decision based on the different condition. You can use conditional statements in your code to make your decisions. PHP supports.

if statement: PHP if else statement is used to test condition. There are various ways to use if statement in PHP like simple if, if-else, if-else-if, and nested if. PHP simple if statement allows conditional execution of code. It is executed if condition is true. If statement is used to executes the block of code exist inside the if statement only if the specified condition is true. Syntax:

```
	if(condition){
		//code to be executed
	}
```

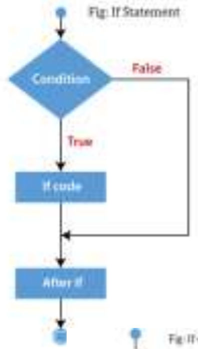


Fig: If Statement

PHP if-else statement is executed whether condition is true or false. If-else statement is slightly different from if statement. It executes one block of code if the specified condition is true and another block of code if the condition is false. Syntax:

```
	if(condition){
		//code to be executed if true
	}else{
		//code to be executed if false
	}
```

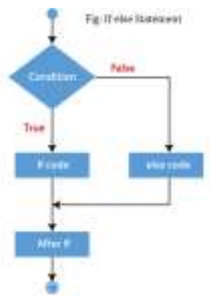


Fig: If-else Statement

The PHP if-else-if is a special statement used to combine multiple if-else statements. So, we can check multiple conditions using this statement. Syntax

```
	if (condition1){
		//code to be executed if C1 is true
	} elseif (condition2){
		//code to be executed if C2 is true
	} elseif (condition3){
		//code to be executed if C3 is true
	}  else{
		//code to be executed if all are
```

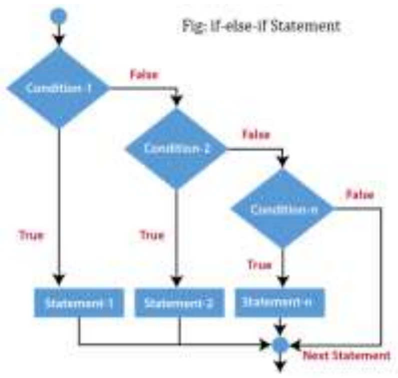
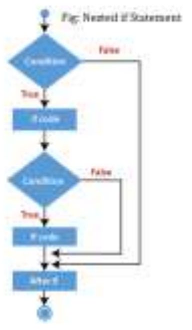


Fig: If-else-if Statement

```
          false
     }
```

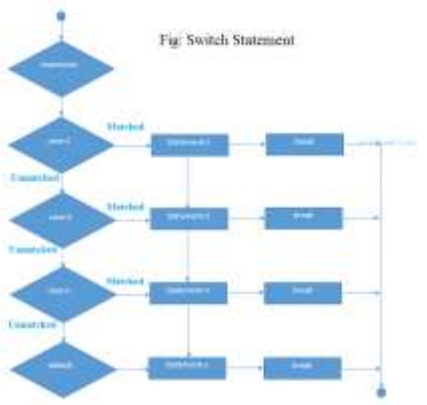
The nested if statement contains the if block inside another if block. The inner if statement executes only when specified condition in outer if statement is true. Syntax:

```
     if (condition) {
          //code to be executed if condition is true
          if (condition) {
               //code to be executed if condition is true
          }
     }
```



Switch…case statement: PHP switch statement is used to execute one statement from multiple conditions. It works like PHP if-else-if statement. Syntax:

```
     switch(expression){
          case value1:
             //code to be executed
             break;
          case value2:
             //code to be executed
             break;
          default:
             //executed if cases are not matched;
     }
```



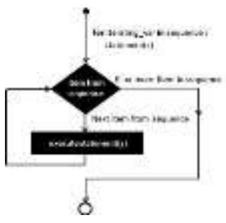
Important points to be noticed about switch case: The default is an optional statement. Even it is not important, that default must always be the last statement. There can be only one default in a switch statement. More than one default may lead to a Fatal error. Each case can have a break statement, which is used to terminate the sequence of statement. The break statement is optional to use in switch. If break is not used, all the statements will execute after finding matched case value. PHP allows you to use number, character, string, as well as functions in switch expression. Nesting of switch statements is allowed, but it makes the program more complex and less readable. You can use semicolon (;) instead of colon (:). It will not generate any error.

Looping Structure

Loops in PHP are used to execute the same block of code a specified number of times. PHP supports following four loop types. **for − loops** through a block of code a specified number of times. **while − loops** through a block of code if and as long as a specified condition is true. **do...while** − loops through a block of code once, and then repeats the loop as long as a special condition is true. **foreach − loops** through a block of code for each element in an array. We will discuss about **continue** and **break** keywords used to control the loops execution.

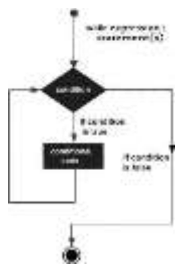
for loop statement: The for statement is used when you know how many times you want to execute a statement or a block of statements. Syntax:

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



The initializer is used to set the start value for the counter of the number of loop iterations. A variable may be declared here for this purpose and it is traditional to name it $i.

while loop statement: The while statement will execute a block of code if and as long as a test expression is true. If the test expression is true then the code block will be executed. After the code has executed the test expression will again be evaluated and the loop will continue until the test expression is found to be false. Syntax:



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

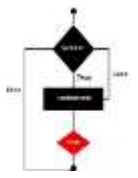
do...while loop statement: The do...while statement will execute a block of code at least once - it then will repeat the loop as long as a condition is true. Syntax:

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

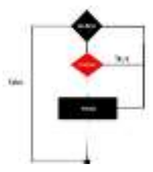
foreach loop statement: The foreach statement is used to loop through arrays. For each pass the value of the current array element is assigned to $value and the array pointer is moved by one and in the next pass next element will be processed. Syntax:

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

break statement: The PHP break keyword is used to terminate the execution of a loop prematurely. The break statement is situated inside the statement block. It gives you full control and whenever you want to exit from the loop you can come out. After coming out of a loop immediate statement to the loop will be executed.



continue statement: The PHP continue keyword is used to halt the current iteration of a loop but it does not terminate the loop. Just like the break statement the continue statement is situated inside the statement block containing the code that the loop executes, preceded by a conditional test. For the pass encountering continue statement, rest of the loop code is skipped and next pass starts.



Array

An array is a data structure that stores one or more similar type of values in a single value. For example if you want to store 100 numbers then instead of defining 100 variables its easy to define an array of 100 length. There are three different kind of arrays and each array value is accessed using an ID c which is called array index. Numeric array − An array with a numeric index. Values are stored and accessed in linear fashion. Associative array − An array with strings as index. This store element values in association with key values rather than in a strict linear index order. Multidimensional array − An array containing one or more arrays and values are accessed using multiple indices.

Numeric Array (Indexed Array): These arrays can store numbers, strings and any object but their index will be represented by numbers. By default, array index starts from zero. Following is the example showing how to create and access numeric arrays:

Here we have used array() function to create array. This function is explained in function reference.

```
<html>
   <body>

      <?php
         /* First method to create array. */
         $numbers = array( 1, 2, 3, 4, 5);

         foreach( $numbers as $value ) {
            echo "Value is $value <br />";
         }

         /* Second method to create array. */
         $numbers[0] = "one";
         $numbers[1] = "two";
         $numbers[2] = "three";
         $numbers[3] = "four";
         $numbers[4] = "five";

         foreach( $numbers as $value ) {
            echo "Value is $value <br />";

         }
      ?>

   </body>
</html>
```

This will produce the following result −

```
Value is 1
Value is 2
Value is 3
Value is 4
Value is 5
Value is one
Value is two
Value is three
Value is four
Value is five
```

Associative Arrays: The associative arrays are very similar to numeric arrays in term of functionality but they are different in terms of their index. Associative array will have their index as string so that you can establish a strong association between key and values. To store the salaries of employees in an array, a numerically indexed array would not be the best choice. Instead, we could use the employees names as the keys in our associative array, and the value would be their respective salary. Example:

```
<html>
   <body>
      <?php
         /* First method to associate create array. */
         $salaries = array("mohammad" => 2000, "qadir" => 1000, "zara" => 500);
         echo "Salary of mohammad is ". $salaries['mohammad'] . "<br />";
         echo "Salary of qadir is ".  $salaries['qadir']. "<br />";
         echo "Salary of zara is ".  $salaries['zara']. "<br />";
         /* Second method to create array. */
         $salaries['mohammad'] = "high";
         $salaries['qadir'] = "medium";
         $salaries['zara'] = "low";
         echo "Salary of mohammad is ". $salaries['mohammad'] . "<br />";
         echo "Salary of qadir is ".  $salaries['qadir']. "<br />";
         echo "Salary of zara is ".  $salaries['zara']. "<br />";
      ?>

   </body>
</html>
```

This will produce the following result −

```
Salary of mohammad is 2000
Salary of qadir is 1000
Salary of zara is 500
Salary of mohammad is high
Salary of qadir is medium
Salary of zara is low
```

Multidimensional Arrays: A multi-dimensional array each element in the main array can also be an array. And each element in the sub-array can be an array, and so on. Values in the multi-dimensional array are accessed using multiple index.

User Defined Functions

PHP functions are similar to other programming languages. A function is a piece of code which takes one more input in the form of parameter and does some processing and returns a value. You already have seen many functions like fopen() and fread() etc. They are built-in functions but PHP gives you option to create your own functions as well. There are two parts which should be clear to you: Creating a PHP Function, and Calling a PHP Function.

In fact you hardly need to create your own PHP function because there are already more than 1000 of built-in library functions created for different area and you just need to call them according to your requirement. Please refer to PHP Function Reference for a complete set of useful functions.

Creating PHP Function: Its very easy to create your own PHP function. Suppose you want to create a PHP function which will simply write a simple message on your browser when you will call it. Following example creates a function called writeMessage() and then calls it just after creating it. Note that while creating a function its name should start with keyword function and all the PHP code should be put inside { and } braces as shown in the following example below:

```
<html>

   <head>
      <title>Writing PHP Function</title>
   </head>

   <body>

      <?php
         /* Defining a PHP Function */
         function writeMessage() {
            echo "You are really a nice person, Have a nice time!";
         }

         /* Calling a PHP Function */
         writeMessage();
      ?>

   </body>
</html>
```

This will display following result –

```
You are really a nice person, Have a nice time!
```

PHP Functions with Parameters: PHP gives you option to pass your parameters inside a function. You can pass as many as parameters your like. These parameters work like variables inside your function. Following example takes two integer parameters and add them together and then print them.

```
<html>
   <head>
      <title>Writing PHP Function with Parameters</title>
   </head>
   <body>

      <?php
         function addFunction($num1, $num2) {
            $sum = $num1 + $num2;
            echo "Sum of the two numbers is : $sum";
         }
         addFunction(10, 20);
      ?>
   </body>
</html>
```

This will display following result –

```
Sum of the two numbers is: 30
```

Passing Arguments by Reference: It is possible to pass arguments to functions by reference. This means that a reference to the variable is manipulated by the function rather than a copy of the variable's value. Any changes made to an argument in these cases will change the value of the original variable. You can pass an argument by reference by adding an ampersand to the variable name in either the function call or the function definition. Following example depicts both the cases.

```
<html>
   <head>
      <title>Passing Argument by Reference</title>
   </head>
   <body>
      <?php
         function addFive($num) {
            $num += 5;
         }

         function addSix(&$num) {
            $num += 6;
         }

         $orignum = 10;
         addFive( $orignum );
         echo "Original Value is $orignum<br />";
         addSix( $orignum );
         echo "Original Value is $orignum<br />";
      ?>
   </body>
</html>
```

This will display following result –

```
Original Value is 10
Original Value is 16
```

PHP Functions returning value: A function can return a value using the return statement in conjunction with a value or object. return stops the execution of the function and sends the value back to the calling code. You can return more than one value from a function using return array(1,2,3,4). Following example takes two integer parameters and add them together and then returns their sum to the calling program. Note that return keyword is used to return a value from a function.

```
<html>
   <head>
      <title>Writing PHP Function which returns value</title>
   </head>
   <body>
      <?php
         function addFunction($num1, $num2) {
            $sum = $num1 + $num2;
            return $sum;
         }
         $return_value = addFunction(10, 20);
         echo "Returned value from the function : $return_value";
      ?>
   </body>
</html>
```

This will display following result –

```
Returned value from the function: 30
```

Setting Default Values for Function Parameters: You can set a parameter to have a default value if the function's caller doesn't pass it. Following function prints NULL in case use does not pass any value to this function.

```
<html>
   <head>
      <title>Writing PHP Function which returns value</title>
   </head>
   <body>

      <?php
         function printMe($param = NULL) {
            print $param;
         }
         printMe("This is test");
         printMe();
      ?>
   </body>
</html>
```

This will produce following result −

```
This is test
```

Dynamic Function Calls: It is possible to assign function names as strings to variables and then treat these variables exactly as you would the function name itself. Following example depicts this behaviour.

```
<html>
   <head>
      <title>Dynamic Function Calls</title>
   </head>
   <body>
      <?php
         function sayHello() {
            echo "Hello<br />";
         }
         $function_holder = "sayHello";
         $function_holder();
      ?>
   </body>
</html>

```

This will display following result −

Hello

Variable Length Argument Function

func_get_arg()	This function can return an item from an argument list.
func_get_args()	This function can return an array comprising a function's argument list.
func_num_args()	This function can return the number of arguments passed to a function.

Built in Functions

- **Variable Functions:** PHP Variable Handling Functions are the inbuilt PHP library functions that enable us to manipulate and test php variables in various ways. There is no installation needed to use PHP variable handling functions; they are part of the PHP core and comes along with standard PHP installation. This extension has no configuration directives defined in php.ini. Function like **gettype(), is_array(), is_bool(), is_null(), empty(), and unset()** etc.

- **String Function:** PHP string functions are the part of the core. There is no installation required to use this function. Functions like **fprintf(), join(), print(), printf(), strlen(), strcmp(), and substr()** etc.
- **Math Function:** PHP provides many predefined math constants and functions that can be used to perform mathematical operations. Functions like **abs(), ceil(), floor(), sqrt(), decbin(), cos(), and log10()** etc.

- **Date Function:** These functions allow you to get the date and time from the server where your PHP scripts are running. You can use these functions to format the date and time in many different ways. There is no installation needed to use these functions; they are part of the PHP core. The behavior of the these functions is affected by settings in php.ini. All these parameters are available in PHP version 5 and onwards. Functions like **date_format(), date(), getdate(), time(), date_add(), date_diff(), date_sub(), and date_time_set()** etc.

- **Array Function:** PHP Array Functions allow you to interact with and manipulate arrays in various ways. PHP arrays are essential for storing, managing, and operating on sets of variables. PHP supports simple and multi-dimensional arrays and may be either user created or created by another function. There is no installation needed to use PHP array functions; they are part of the PHP core and comes along with standard PHP installation. This extension has no configuration directives defined in php.ini. function like **array(), array_diff(), array_map(), array_merge(), array_slice(), asort(), and arsort()** etc.

- **File handling Function:** The file system functions are used to access and manipulate the file system PHP provides you all the possible functions you may need to manipulate a file. The error and logging functions are part of the PHP core. There is no installation needed to use these functions. Functions like **fopen(), fclose(), feof(), filesize(), delete(), copy(), and file_exist()** etc.

Unit – 2

Handling Form Session Tracking & PHP Components

AJAX & JSON

Handling form with GET & POST

(see Unit-2 GET & POST method topic)

Cookies

Cookies are text files stored on the client computer and they are kept of use tracking purpose. PHP transparently supports HTTP cookies. There are three steps involved in identifying returning users −

- Server script sends a set of cookies to the browser. For example name, age, or identification number etc.
- Browser stores this information on local machine for future use.
- When next time browser sends any request to web server then it sends those cookies information to the server and server uses that information to identify the user.

The following paragraphs will teach you how to set cookies, how to access them and how to delete them. Cookies are usually set in an HTTP header (although JavaScript can also set a cookie directly on a browser). As you can see, the Set-Cookie header contains a name value pair, a GMT date, a path and a domain. The name and value will be URL encoded. The expires field is an instruction to the browser to "forget" the cookie after the given time and date.

If the browser is configured to store cookies, it will then keep this information until the expiry date. If the user points the browser at any page that matches the path and domain of the cookie, it will resend the cookie to the server.

Setting Cookies with PHP: PHP provided setcookie() function to set a cookie. This function requires upto six arguments and should be called before <html> tag. For each cookie this function has to be called separately.

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

Here is the detail of all the arguments −

- Name − This sets the name of the cookie and is stored in an environment variable called HTTP_COOKIE_VARS. This variable is used while accessing cookies.
- Value − This sets the value of the named variable and is the content that you actually want to store.
- Expiry − This specify a future time in seconds since 00:00:00 GMT on 1st Jan 1970. After this time cookie will become inaccessible. If this parameter is not set then cookie will automatically expire when the Web Browser

is closed.

- Path − This specifies the directories for which the cookie is valid. A single forward slash character permits the cookie to be valid for all directories.
- Domain − This can be used to specify the domain name in very large domains and must contain at least two periods to be valid. All cookies are only valid for the host and domain which created them.
- Security − This can be set to 1 to specify that the cookie should only be sent by secure transmission using HTTPS otherwise set to 0 which mean cookie can be sent by regular HTTP.

Following example will create two cookies name and age these cookies will be expired after one hour.

```
<?php
   setcookie("name", "Dr. Malay B. Dave", time()+3600, "/","", 0);
   setcookie("age", "40", time()+3600, "/", "",  0);
?>
```

Accessing Cookies with PHP: PHP provides many ways to access cookies. Simplest way is to use either $_COOKIE or $HTTP_COOKIE_VARS variables. Following example will access all the cookies set in above example.

```
<?php
   echo $_COOKIE["name"]. "<br />";
?>
```

Deleting Cookie with PHP: Officially, to delete a cookie you should call setcookie() with the name argument only but this does not always work well, however, and should not be relied on. It is safest to set the cookie with a date that has already expired.

Session

An alternative way to make data accessible across the various pages of an entire website is to use a PHP Session. A session creates a file in a temporary directory on the server where registered session variables and their values are stored. This data will be available to all pages on the site during that visit. The location of the temporary file is determined by a setting in the php.ini file called session.save_path. Before using any session variable make sure you have setup this path. When a session is started following things happen −

- PHP first creates a unique identifier for that particular session which is a random string of 32 hexadecimal numbers such as 3c7foj34c3jj973hjkop2fc937e3443.
- A cookie called PHPSESSID is automatically sent to the user's computer to store unique session identification string.
- A file is automatically created on the server in the designated temporary directory and bears the name of the unique identifier prefixed by sess_ ie sess_3c7foj34c3jj973hjkop2fc937e3443.

When a PHP script wants to retrieve the value from a session variable, PHP automatically gets the unique session identifier string from the PHPSESSID cookie and then looks in its temporary directory for the file bearing that name and a validation can be done by comparing both values. A session ends when the user loses the browser or after leaving the site, the server will terminate the session after a predetermined period of time, commonly 30 minutes duration.

Starting a PHP Session: A PHP session is easily started by making a call to the **session_start()** function. This function first checks if a session is already started and if none is started then it starts one. It is recommended to put the call to session_start() at the beginning of the page. Session variables are stored in associative array called **$_SESSION[].** These variables can be accessed during lifetime of a session. The following example starts a session then register a variable called counter that is incremented each time the page is visited during the session.

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

```
                $_SESSION["user"] = "Sachin";
                echo "Session information are set successfully.<br/>";
        ?>
        </body>
    </html>
```

PHP Destroying Session: PHP **session_destroy()** function is used to destroy all session variables completely.

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

Server variable

$_SERVER is an array containing information such as headers, paths, and script locations. The entries in this array are created by the web server. There is no guarantee that every web server will provide any of these.

Sr.	Variable & Description
1	**$_SERVER['PHP_SELF']:** The filename of the currently executing script, relative to the document root
2	**$_SERVER['argv']:** Array of arguments passed to the script. When the script is run on the command line, this gives C-style access to the command line parameters. When called via the GET method, this will contain the query string.
3	**$_SERVER['argc']:** Contains the number of command line parameters passed to the script if run on the command line.
4	**$_SERVER['GATEWAY_INTERFACE']:** What revision of the CGI specification the server is using; i.e. 'CGI/1.1'.
5	**$_SERVER['SERVER_ADDR']:**The IP address of the server under which the current script is executing.
6	**$_SERVER['SERVER_NAME']:** The name of the server host under which the current script is executing. If the script is running on a virtual host, this will be the value defined for that virtual host.
7	**$_SERVER['SERVER_SOFTWARE']:** Server identification string, given in the headers when responding to requests.
8	**$_SERVER['SERVER_PROTOCOL']:** Name and revision of the information protocol via which the page was requested; i.e. 'HTTP/1.0';
9	**$_SERVER['REQUEST_METHOD']:** Which request method was used to access the page; i.e. 'GET', 'HEAD', 'POST', 'PUT'.
10	**$_SERVER['REQUEST_TIME']:** The timestamp of the start of the request. Available since PHP 5.1.0.

PHP Components

PHP GD (Graphic Display) Library: PHP is not limited to creating just HTML output. It can also be used to create and manipulate image files in a variety of different image formats, including GIF, PNG, JPEG, WBMP, and XPM. Even more conveniently, PHP can output image streams directly to a browser. You will need to compile PHP with the GD library of image functions for this to work. GD and PHP may also require other libraries, depending on which image formats you want to work with. You can use the image functions in PHP to get the size of JPEG, GIF, PNG, SWF, TIFF and JPEG2000 images. With the exif extension, you are able to work with information stored in headers of JPEG and TIFF images. This way you can read meta data generated by digital cameras. The exif functions do not require the GD library. GD supports a varity of formats, below is a list of formats supported by GD and notes to their availability including read/write support.

Format	Read support	Write support
JPEG	`true`	`true`
PNG	`true`	`true`
GIF	`true`	`true`

Format	Read support	Write support
XBM	`true`	`true`
XPM	`true`	`false`
WBMP	`true`	`true`
WebP	`true`	`true`
BMP	`true`	`true`

Despite most formats being available for both reading and writing in the above table, doesn't mean that PHP was compiled with support for them. To find out which formats that was available to GD during compilation, use the gd_info() function, for more information about compiling support for one or more formats. GD and Image function list:

- gd_info — Retrieve information about the currently installed GD library
- getimagesize — Get the size of an image
- getimagesizefromstring — Get the size of an image from a string
- image_type_to_extension — Get file extension for image type
- image_type_to_mime_type — Get Mime-Type for image-type returned by getimagesize, exif_read_data, exif_thumbnail, exif_imagetype
- image2wbmp — Output image to browser or file
- imageaffine — Return an image containing the affine transformed src image, using an optional clipping area
- imageaffinematrixconcat — Concatenate two affine transformation matrices
- imageaffinematrixget — Get an affine transformation matrix
- imagealphablending — Set the blending mode for an image
- imageantialias — Should antialias functions be used or not
- imagearc — Draws an arc
- imageavif — Output image to browser or file
- imagebmp — Output a BMP image to browser or file
- imagechar — Draw a character horizontally
- imagecharup — Draw a character vertically

PHP Regular expression

Regular expressions are nothing more than a sequence or pattern of characters itself. They provide the foundation for pattern-matching functionality. Using regular expression you can search a particular string inside a another string, you can replace one string by another string and you can split a string into many chunks. PHP offers functions specific to two sets of regular expression functions, each corresponding to a certain type of regular expression. You can use any of them based on your comfort. POSIX Regular Expressions and PERL Style Regular Expressions.

POSIX Regular Expressions: The structure of a POSIX regular expression is not dissimilar to that of a typical arithmetic expression: various elements (operators) are combined to form more complex expressions. The simplest regular expression is one that matches a single character, such as g, inside strings such as g, haggle, or bag. Lets give explanation for few concepts being used in POSIX regular expression. After that we will introduce you with regular expression related functions.

Brackets: Brackets ([]) have a special meaning when used in the context of regular expressions. They are used to find a range of characters.

Sr.	Expression & Description
1	**[0-9]:** It matches any decimal digit from 0 through 9.
2	**[a-z]:** It matches any character from lower-case a through lowercase z.
3	**[A-Z]:** It matches any character from uppercase A through uppercase Z.
4	**[a-Z]:** It matches any character from lowercase a through uppercase Z.

The ranges shown above are general; you could also use the range [0-3] to match any decimal digit ranging from 0 through 3, or the range [b-v] to match any lowercase character ranging from b through v.

Quantifiers: The frequency or position of bracketed character sequences and single characters can be denoted by a special character. Each special character having a specific connotation. The +, *, ?, {int. range}, and $ flags all follow a character sequence.

Sr.	Expression & Description
1	**p+ :** It matches any string containing at least one p.
2	**p* :** It matches any string containing zero or more p's.
3	**p? :** It matches any string containing zero or one p's.
4	**p{N} :** It matches any string containing a sequence of **N** p's
5	**p{2,3} :** It matches any string containing a sequence of two or three p's.
6	**p{2, } :** It matches any string containing a sequence of at least two p's.
7	**p$:** It matches any string with p at the end of it.
8	**^p :** It matches any string with p at the beginning of it.

Following examples will clear your concepts about matching characters.

Sr.	Expression & Description
1	**[^a-zA-Z] :** It matches any string not containing any of the characters ranging from a through z and A through Z.
2	**p.p :** It matches any string containing p, followed by any character, in turn followed by another p.
3	**^.{2}$:** It matches any string containing exactly two characters.
4	**<b>(.*)</b> :** It matches any string enclosed within <b> and </b>.
5	**p(hp)* :** It matches any string containing a p followed by zero or more instances of the sequence php.

Predefined Character Ranges: For your programming convenience several predefined character ranges, also known as character classes, are available. Character classes specify an entire range of characters, for example, the alphabet or an integer set –

Sr.	Expression & Description
1	**[[:alpha:]] :** It matches any string containing alphabetic characters aA through zZ.
2	**[[:digit:]] :** It matches any string containing numerical digits 0 through 9.
3	**[[:alnum:]] :** It matches any string containing alphanumeric characters aA through zZ and 0 through 9.
4	**[[:space:]] :** It matches any string containing a space.

PHP's Regexp POSIX Functions: PHP currently offers seven functions for searching strings using POSIX-style regular expressions –

Sr.	Function & Description
1	**ereg() :** The ereg() function searches a string specified by string for a string specified by pattern, returning true if the pattern is found, and false otherwise.
2	**ereg_replace() :** The ereg_replace() function searches for string specified by pattern and replaces pattern with replacement if found.
3	**eregi() :** The eregi() function searches throughout a string specified by pattern for a string specified by string. The search is not case sensitive.
4	**eregi_replace() :** The eregi_replace() function operates exactly like ereg_replace(), except that the search for pattern in string is not case sensitive.
5	**split() :** The split() function will divide a string into various elements, the boundaries of each element based on the occurrence of pattern in string.
6	**spliti() :** The spliti() function operates exactly in the same manner as its sibling split(), except that it is not case sensitive.
7	**sql_regcase() :** The sql_regcase() function can be thought of as a utility function, converting each character in the input parameter string into a bracketed expression containing two characters.

Uploading file
PHP allows you to upload single and multiple files through few lines of code only. PHP file upload features allows you to upload binary and text files both. Moreover, you can have the full control over the file to be uploaded through PHP authentication and file operation functions.

PHP $_FILES: The PHP global $_FILES contains all the information of file. By the help of $_FILES global, we can get file name, file type, file size, temp file name and errors associated with file. Here, we are assuming that file name is filename.

$_FILES['filename']['name']: returns file name.
$_FILES['filename']['type']: returns MIME type of the file.
$_FILES['filename']['size']: returns size of the file (in bytes).
$_FILES['filename']['tmp_name']: returns temporary file name of the file which was stored on the server.
$_FILES['filename']['error']: returns error code associated with this file.
move_uploaded_file() function: The move_uploaded_file() function moves the uploaded file to a new location. The move_uploaded_file() function checks internally if the file is uploaded thorough the POST request. It moves the file if it is uploaded through the POST request. Syntax:

```
  bool move_uploaded_file ( string $filename , string $destination )
```

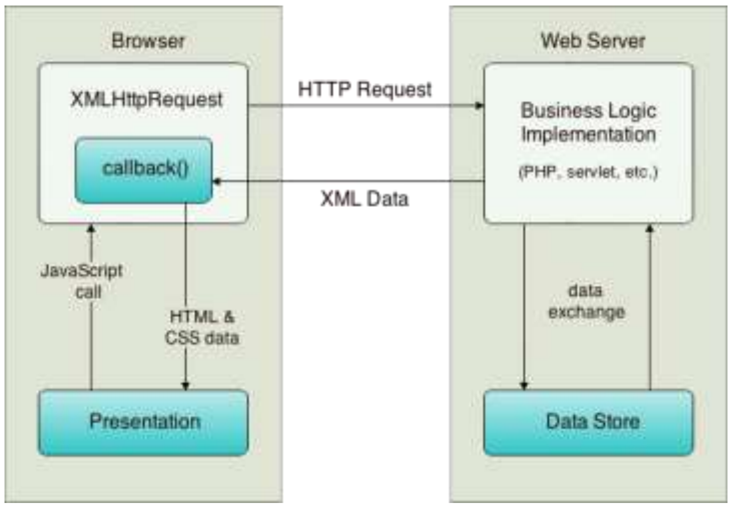
Sending mail
PHP **mail()** function is used to send email in PHP. You can send text message, html message and attachment with message using PHP mail() function. PHP mail() function’s Syntax:

```
   bool mail ( string $to , string $subject , string $message [, string
$additional_headers [, string $additional_parameters ]] )
```

$to: specifies receiver or receivers of the mail. The receiver must be specified one of the following forms.
user@example.com
user@example.com, anotheruser@example.com
User <user@example.com>
User <user@example.com>, Another User <anotheruser@example.com>
$subject: represents subject of the mail.
$message: represents message of the mail to be sent. Each line of the message should be separated with a CRLF (\r\n) and lines should not be larger than 70 characters.
$additional_headers (optional): specifies the additional headers such as From, CC, BCC etc. Extra additional headers should also be separated with CRLF (\r\n).

AJAX

What is AJAX? AJAX stands for **Asynchronous JavaScript and XML**. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS and Java Script. Conventional web application transmit information to and from the sever using synchronous requests. This means you fill out a form, hit submit, and get directed to a new page with new information from the server. With AJAX when submit is pressed, JavaScript will make a request to the server, interpret the results and update the current screen. In the purest sense, the user would never know that anything was even transmitted to the server. AJAX cannot work independently. It is used in combination with other technologies to create interactive webpages. AJAX communicates with the server using XMLHttpRequest object. Let's try to understand the flow of ajax or how ajax works by the image displayed below.



As you can see in the above example, XMLHttpRequest object plays a important role. User sends a request from the UI and a javascript call goes to XMLHttpRequest object. HTTP Request is sent to the server by XMLHttpRequest object. Server interacts with the database using JSP, PHP, Servlet, ASP.net etc. Data is retrieved. Server sends XML data or JSON data to the XMLHttpRequest callback function. HTML and CSS data is displayed on the browser.

- **JavaScript:** Loosely typed scripting language. JavaScript function is called when an event occurs in a page. Glue for the whole AJAX operation.
- **DOM:** API for accessing and manipulating structured documents. Represents the structure of XML and HTML documents.
- **CSS:** Allows for a clear separation of the presentation style from the content and may be changed programmatically by JavaScript
- **XMLHttpRequest:** JavaScript object that performs asynchronous interaction with the server.

Unit – 3

Introduction of SQL

Working with MySQL using PhpMyAdmin

MySQL is currently the most popular database management system software used for managing the relational database. It is open-source database software, which is supported by Oracle Company. It is fast, scalable, and easy to use database management system in comparison with Microsoft SQL Server and Oracle Database. It is commonly used in conjunction with PHP scripts for creating powerful and dynamic server-side or web-based enterprise applications. It is developed, marketed, and supported by **MySQL AB**, a Swedish company, and written in C programming language and C++ programming language. The official pronunciation of MySQL is not the My Sequel; it is My Ess Que Ell. However, you can pronounce it in your way. Many small and big companies use MySQL. MySQL supports many Operating Systems like Windows.

MySQL is a Relational Database Management System (RDBMS) software that provides many things, which are as: It allows us to implement database operations on tables, rows, columns, and indexes. It defines the database relationship in the form of tables (collection of rows and columns), also known as relations. It provides the Referential Integrity between rows or columns of various tables. It allows us to updates the table indexes automatically. It uses many SQL queries and combines useful information from multiple tables for the end-users.

phpMyAdmin is a free software tool written in PHP, intended to handle the administration of MySQL over the Web. phpMyAdmin supports a wide range of operations on MySQL and MariaDB. Frequently used operations (managing databases, tables, columns, relations, indexes, users, permissions, etc.) can be performed via the user interface, while you still have the ability to directly execute any SQL statement.

SQL DML Statement

Insert: You can add new rows to an existing table of MySQL using the INSERT statement. In this, you need to specify the name of the table, column names, and values (in the same order as column names). Following is the syntax of the INSERT statement of MySQL.

```
INSERT INTO table_name (column1, column2, column3,...columnN)
VALUES (value1, value2, value3,...valueN);
```

Where, table_name is the name of the table into which you need to insert data, (column1, column2, column3,...columnN) are the names of the columns and (value1, value2, value3,...valueN) are the values in the record. Assume we have created a table with name Sales in MySQL database using CREATE TABLE statement as shown below:

```
CREATE TABLE sales(
   ID INT,
   ProductName VARCHAR(255),
   CustomerName VARCHAR(255),
   DispatchDate date,
   DeliveryTime time,
   Price INT,
   Location VARCHAR(255)
);
```

Update: There may be a requirement where the existing data in a MySQL table needs to be modified. You can do so by using the SQL UPDATE command. This will modify any field value of any MySQL table. The following code block has a generic SQL syntax of the UPDATE command to modify the data in the MySQL table −

```
UPDATE table_name SET field1 = new-value1, field2 = new-value2
[WHERE Clause]
```

You can update one or more field altogether. You can specify any condition using the WHERE clause. You can update the values in a single table at a time. The WHERE clause is very useful when you want to update the selected rows in a table. The following example will update the tutorial_title field for a record having the tutorial_id as 3.

```
UPDATE tutorials_tbl
   SET tutorial_title = 'Learning PHP'
   WHERE tutorial_id = 3;
```

Select: The SQL SELECT command is used to fetch data from the MySQL database. You can use this command at mysql> prompt as well as in any script like PHP. Here is generic SQL syntax of SELECT command to fetch data from the MySQL table −

```
SELECT field1, field2,...fieldN
FROM table_name1, table_name2...
[WHERE Clause]
```

You can use one or more tables separated by comma to include various conditions using a WHERE clause, but the WHERE clause is an optional part of the SELECT command. You can fetch one or more fields in a single SELECT command. You can specify star (*) in place of fields. In this case, SELECT will return all the fields. You can specify any condition using the WHERE clause. You can specify an offset using OFFSET from where SELECT will start returning records. By default, the offset starts at zero. The following example will return all the records from the tutorials_tbl table −

```
SELECT * from tutorials_tbl;
```

Delete: If you want to delete a record from any MySQL table, then you can use the SQL command DELETE FROM. You can use this command at the mysql> prompt as well as in any script like PHP. The following code block has a generic SQL syntax of the DELETE command to delete data from a MySQL table.

```
DELETE FROM table_name [WHERE Clause]
```

If the WHERE clause is not specified, then all the records will be deleted from the given MySQL table. You can specify any condition using the WHERE clause. You can delete records in a single table at a time. The WHERE clause is very useful when you want to delete selected rows in a table. The following example will delete a record from the tutorial_tbl whose tutorial_id is 3.

```
DELETE FROM tutorials_tbl WHERE tutorial_id=3;
```

PHP-MySQLi Connectivity

PHP 5 and later can work with a MySQL database using MySQLi extension (the "i" stands for improved). Earlier versions of PHP used the MySQL extension. However, this extension was deprecated in 2012. MySQLi will only work with MySQL databases.

PHP-MySQLi Functions

mysqli_connect(): PHP provides mysqli contruct or mysqli_connect() function to open a database connection. This function takes six parameters and returns a MySQL link identifier on success or FALSE on failure. Syntax:

```
$mysqli = new mysqli_connect($host, $username, $passwd, $dbName, $port, $socket);
```

- **$host:** Optional − The host name running the database server. If not specified, then the default value will be localhost:3306.

- **$username:** Optional – The username accessing the database. If not specified, then the default will be the name of the user that owns the server process.
- **$passwd:** Optional – The password of the user accessing the database. If not specified, then the default will be an empty password.
- **$dbName:** Optional – database name on which query is to be performed.
- **$port:** Optional – the port number to attempt to connect to the MySQL server.
- **$socket:** Optional – socket or named pipe that should be used.

mysqli_close(): The mysqli_close() function accepts a MySQL function object (previously opened) as a

parameter, and closes it. You cannot close persistent connections using this function. Syntax:

```
    mysqli_close($con);
```

mysqli_error(): It returns the last error description for the most recent statement. The mysqli_error() function returns the description of the error occurred during the last MySQLi function call. Syntax:

```
    mysqli_error($con)
```

msyqli_errno(): It returns the last error code for the most recent statement. The mysqli_errno() function returns the code of the error occurred during the last MySQLi function call. Syntax:

```
    mysqli_errno($con)
```

mysqli_select_db(): This function changes the default database. The mysqli_select_db() function accepts a string value representing an existing database and, makes it as a the default database. Syntax:

```
    mysqli_select_db($con, name)
```

mysqli_query(): It performs a query against the database. The mysqli_query() function accepts a string value representing a query as one of the parameters and, executes/performs the given query on the database. Syntax:

```
    mysqli_query($con, query)
```

mysqli_fetch_array(): It is used to fetchs a result row as an associative array. A PHP result object (of the class mysqli_result) represents the MySQL result, returned by the SELECT or, DESCRIBE or, EXPLAIN queries. The mysqli_fetch_array() function accepts a result object as a parameter and, retrieves the contents of current row in the given result object, and returns them as an associative or, numeric array. Syntax

```
    mysqli_fetch_array($result, [$type]);
```

mysqli_num_row(): It returns the contents of the current row of a result as an array of strings. A PHP result object (of the class mysqli_result) represents the MySQL result, returned by the SELECT or, DESCRIBE or, EXPLAIN queries. The mysqli_num_rows() function accepts a result object as a parameter, retrieves the number of rows in the given result. Syntax:

```
    mysqli_num_rows($result);
```

mysqli_affected_rows(): The mysqli_affected_rows() function returns the number of rows affected by the previous operation, if invoked after INSERT, UPDATE, REPLACE or DELETE query. When used after select statements this function returns the number of rows. Syntax:

```
    mysqli_affected_rows($con)
```

mysqli_fetch_assoc(): It is used to fetches a result row as an associative array. A PHP result object (of the class mysqli_result) represents the MySQL result, returned by the SELECT or, DESCRIBE or, EXPLAIN queries. The mysqli_fetch_assoc() function accepts a result object as a parameter and, retrieves the contents of current row in the given result object, and returns them as an associative array. Syntax:

```
    mysqli_fetch_assoc($result);
```

mysqli_fetch_field(): It is used to returns the next column in the result set as an object. The mysqli_fetch_field() function accepts a result object as a parameter and returns the definition information of the next column/field in the form of an object. Syntax:

```
    mysqli_fetch_field($result);
```

mysqli_fetch_object(): It returns an object. The mysqli_fetch_object() function accepts a result object as a parameter and, retrieves the contents of current row in the given result and returns them as an object. Syntax:

```
    mysqli_fetch_object($result, [$class_name, $params]);
```

mysqli_fetch_row(): It returns the contents of the current row of a result as an array of strings. The mysqli_fetch_row() function accepts a result object as a parameter, retrieves the contents of its current row as an array of strings. Syntax:

```
    mysqli_fetch_row($result);
```

mysqli_insert_id(): It returns an id of last query. If you have a table with an AUTO_INCREMENT attribute for a column and, if your last MySQLi function call executes INSERT or UPDATE statement. The mysqli_insert_id() function returns the auto generated id of the last executed query. Syntax:

```
    mysqli_insert_id($con)
```

mysqli_num_fields(): It returns the number of fields in a result set. The mysqli_num_fields() function accepts a result object as a parameter, retrieves and returns the number of fields in the given object. Syntax:

```
    mysqli_num_fields($result);
```

mysqli_data_seek(): It is used to move internal result pointer. The mysqli_data_seek() function accepts a result object and an integer value representing an offset, as parameters, and moves the data seek of the given result object to the specified row. Syntax:

```
    mysqli_data_seek($result, $offset);
```

Unit – 4
jQuery

What is jQuery?
jQuery is a lightweight, "write less, do more", JavaScript library. jQuery is a fast and concise JavaScript Library created by John Resig in 2006 with a nice motto: Write less, do more. jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is a JavaScript toolkit designed to simplify various tasks by writing less code. Here is the list of important core features supported by jQuery:

- **DOM manipulation** − The jQuery made it easy to select DOM elements, negotiate them and modifying their content by using cross-browser open source selector engine called Sizzle.
- **Event handling** − The jQuery offers an elegant way to capture a wide variety of events, such as a user clicking on a link, without the need to clutter the HTML code itself with event handlers.
- **AJAX Support** − The jQuery helps you a lot to develop a responsive and feature rich site using AJAX technology.
- **Animations** − The jQuery comes with plenty of built-in animation effects which you can use in your websites.
- **Lightweight** − The jQuery is very lightweight library - about 19KB in size (Minified and gzipped).
- **Cross Browser Support** − The jQuery has cross-browser support, and works well in IE 6.0+, FF 2.0+, Safari 3.0+, Chrome and Opera 9.0+
- **Latest Technology** − The jQuery supports CSS3 selectors and basic XPath syntax.

There are two ways to use jQuery. **Local Installation** − You can download jQuery library on your local machine and include it in your HTML code. **CDN Based Version** − You can include jQuery library into your HTML code directly from Content Delivery Network (CDN).

jQuery Syntax
With jQuery you select (query) HTML elements and perform "actions" on them. The jQuery syntax is tailor-made for selecting HTML elements and performing some action on the element(s). Basic syntax is:

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

A $ sign to define/access jQuery. A (selector) to "query (or find)" HTML elements. A jQuery action() to be performed on the element(s). Examples:

```
$(this).hide() - hides the current element.
$("p").hide() - hides all <p> elements.
```

jQuery Selector
jQuery selectors are one of the most important parts of the jQuery library. jQuery selectors allow you to select and manipulate HTML element(s). jQuery selectors are used to "find" (or select) HTML elements based on their name, id, classes, types, attributes, values of attributes and much more. It's based on the existing CSS Selectors, and in addition, it has some own custom selectors. All selectors in jQuery start with the dollar sign and parentheses: $().

The element Selector: The jQuery element selector selects elements based on the element name. You can select all <p> elements on a page like this:

```
$("p")
```

When a user clicks on a button, all <p> elements will be hidden:

```
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide();
  });
});
```

The .class Selector: The jQuery .class selector finds elements with a specific class. To find elements with a specific class, write a period character, followed by the name of the class:

```
$(".test")
```

When a user clicks on a button, the elements with class="test" will be hidden:

```
$(document).ready(function(){
  $("button").click(function(){
    $(".test").hide();
  });
});
```

The #id Selector: The jQuery #id selector uses the id attribute of an HTML tag to find the specific element. An id should be unique within a page, so you should use the #id selector when you want to find a single, unique element. To find an element with a specific id, write a hash character, followed by the id of the HTML element:

```
$("#test")
```

When a user clicks on a button, the element with id="test" will be hidden:

```
$(document).ready(function(){
  $("button").click(function(){
    $("#test").hide();
  });
});
```

jQuery Events

All the different visitors' actions that a web page can respond to are called events. An event represents the precise moment when something happens. Examples: moving a mouse over an element, selecting a radio button, clicking on an element. The term "fires/fired" is often used with events. Example: "The keypress event is fired, the moment you press a key". Here are some common DOM events:

Mouse Events	Keyboard Events	Form Events	Document/Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

- **click():** The click() method attaches an event handler function to an HTML element. The function is executed when the user clicks on the HTML element.
- **dblclick():** The dblclick() method attaches an event handler function to an HTML element. The function is executed when the user double-clicks on the HTML element.
- **mouseenter():** The mouseenter() method attaches an event handler function to an HTML element. The function is executed when the mouse pointer enters the HTML element.
- **mouseleave():** The mouseleave() method attaches an event handler function to an HTML element. The function is executed when the mouse pointer leaves the HTML element.
- **hover():** The hover() method takes two functions and is a combination of the mouseenter() and mouseleave() methods. The first function is executed when the mouse enters the HTML element, and the second function is executed when the mouse leaves the HTML elemen
- **blur():** The blur() method attaches an event handler function to an HTML form field. The function is executed when the form field loses focus.
- **focus():** The focus() method attaches an event handler function to an HTML form field. The function is executed when the form field gets focus.

In jQuery, most DOM events have an equivalent jQuery method. To assign a click event to all paragraphs on a page, you can do this:

```
$("p").click();
```

The next step is to define what should happen when the event fires. You must pass a function to the event:

```
$("p").click(function(){
  // action goes here!!
});
```

jQuery Effects

jQuery hide() and show(): With jQuery, you can hide and show HTML elements with the hide() and show() methods. Syntax:

```
$(selector).hide(speed,callback);
$(selector).show(speed,callback);
```

The optional speed parameter specifies the speed of the hiding/showing, and can take the following values: "slow", "fast", or milliseconds. The optional callback parameter is a function to be executed after the hide() or show() method completes (you will learn more about callback functions in a later chapter). Example:

```
$("#hide").click(function(){
  $("p").hide();
});

$("#show").click(function(){
  $("p").show();
});
```

jQuery Fading Methods: With jQuery you can fade an element in and out of visibility. jQuery has the fade methods like: fadeIn(), fadeOut(), fadeToggle(), fadeTo(). The jQuery fadeIn() method is used to fade in a hidden element. Syntax:

```
$(selector).fadeIn(speed,callback);
```

jQuery Sliding Methods: With jQuery you can create a sliding effect on elements. jQuery has the slide methods like: slideDown(), slideUp(), slideToggle(). The jQuery slideDown() method is used to slide down an element. Syntax:

```
$(selector).slideDown(speed,callback);
```

The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds. The optional callback parameter is a function to be executed after the sliding completes.

Unit – 5
OOP

What is OOP?

OOP stands for Object-Oriented Programming.

Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.

Object-oriented programming has several advantages over procedural programming:

- OOP is faster and easier to execute
- OOP provides a clear structure for the programs
- OOP helps to keep the PHP code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
- OOP makes it possible to create full reusable applications with less code and shorter development time.

What is Class?
A class is a template for objects, and an object is an instance of class.

```
<?php
class Fruit
{
  // code goes here...
}
?>
```

What is Object?

Classes are nothing without objects! We can create multiple objects from a class. Each object has all the properties and methods defined in the class, but they will have different property values. Objects of a class are created using the new keyword.

```
<?php
class Fruit {
  // Properties
  public $name;
  public $color;

  // Methods
  function set_name($name)
{
    $this->name = $name;
  }
  function get_name() {
    return $this->name;
  }
}

$apple = new Fruit();
$banana = new Fruit();
$apple->set_name('Apple');
$banana->set_name('Banana');
```

```
echo $apple->get_name();
echo "<br>";
echo $banana->get_name();
?>
```

What is Property and Visibility ?

Data members declared inside class are called properties. Property is sometimes referred to as attribute or field. In PHP, a property is qualified by one of the access specifier keywords, **public**, **private** or **protected**.

```
<?php
class myclass{
	private $fname="Kiran";
	public $mname="Pratap";
	static $lname="Singh";
	function dispdata(){
		echo "$this->fname\n";
		echo "$this->mname\n";
		echo myclass::$lname;
	}
}
$obj=new myclass();
$obj->dispdata();
?>
```

What is Constructor?

A constructor allows you to initialize an object's properties upon creation of the object.
If you create a __construct() function, PHP will automatically call this function when you create an object from a class.
Notice that the construct function starts with two underscores (__)!

```
<?php
class Fruit {
  public $name;
  public $color;

  function __construct($name) {
    $this->name = $name;
  }
  function get_name() {
    return $this->name;
  }
}

$apple = new Fruit("Apple");
echo $apple->get_name();
?>
```

What is Destructor?

A destructor is called when the object is destructed or the script is stopped or exited.
If you create a __destruct() function, PHP will automatically call this function at the end of the script.
Notice that the destruct function starts with two underscores (__)!

```
<?php
class Fruit {
  public $name;
  public $color;

  function __construct($name) {
    $this->name = $name;
  }
  function __destruct() {
    echo "The fruit is {$this->name}.";
  }
}

$apple = new Fruit("Apple");
?>
```

What is Inheritance?

Inheritance in OOP = When a class derives from another class.
The child class will inherit all the public and protected properties and methods from the parent class. In addition, it can have its own properties and methods.
An inherited class is defined by using the extends keyword.

```
<?php
class test
{
  public function __construct()
  {
    echo "Hello!";
  }
}
class demo extends test
{
  public function alpha()
 {
    echo "Alpha";
 }
}
$obj = new demo;
$obj -> alpha();
?>
```

What is Scope Resolution Operator?

In PHP, the double colon **::** is defined as **Scope Resolution Operator**. It used when when we want to access constants, properties and methods defined at class level. When referring to these items outside class definition, name of class is used along with scope resolution operator. This operator is also called **Paamayim Nekudotayim**, which in Hebrew means double colon.

```
<?php
class A
{
  const PI=3.142;
  static $x=10;
}
```

```
echo A::PI;
echo A::$x;
$var='A';
echo $var::PI;
echo $var::$x;
?>
```

What is Class Constants ?

Constants cannot be changed once it is declared.
Class constants can be useful if you need to define some constant data within a class.
A class constant is declared inside a class with the const keyword.
Class constants are case-sensitive. However, it is recommended to name the constants in all uppercase letters.
We can access a constant from outside the class by using the class name followed by the scope resolution operator (::) followed by the constant name.

```
<?php
class Goodbye {
  const LEAVING_MESSAGE = "Thank you for visiting W3Schools.com!";
}

echo Goodbye::LEAVING_MESSAGE;
?>
```

What is Autoloading Classes ?

In order to use class defined in another PHP script, we can incorporate it with include or require statements. However, PHP's autoloading feature doesn't need such explicit inclusion. Instead, when a class is used (for declaring its object etc.) PHP parser loads it automatically, if it is registered with **spl_autoload_register()** function. Any number of classes can thus be registered. This way PHP parser gets a laast chance to load class/interface before emitting error.

File Name: test1.php

```
Class test1
{

}
```

File Name: test2.php

```
Class test2
{

}

<?php
spl_autoload_register(function ($class_name) {
  include $class_name . '.php';
});
$obj = new test1();
$obj2 = new test2();
echo "objects of test1 and test2 class created successfully";
?>
```