Introduction:

PHP (Hypertext Preprocessor [previously Personal Homepage]) is a widely-used open-source general-purpose scripting language that is especially suited for web development and can be embedded into HTML. It is popular scripting language that is used to create dynamic Web pages. PHP is supported by all Web servers and widely used with the MySQL database and its file extension is .PHP (.php).

Example:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Example</title>
    </head>
    <body>

        <?php
            echo "Hi, I'm a PHP script!";
        ?>
    </body>
</html>
```

Advantages of PHP

1. **Cross-Platform:** PHP is platform-independent. You don't have to have a particular OS to use it because it runs on every platform, whether it's Mac, Windows, or Linux.
2. **Open Source:** PHP is open source. The original code is made available to everyone who wants to build upon it. This is one of the reasons why one of its frameworks, Laravel, is so popular.
3. **Easy to learn:** PHP is not hard to learn for absolute beginners. You can pick it up pretty if you already have programming knowledge.
4. **PHP syncs with all Databases:** You can easily connect PHP to all Databases, relational and non-relational. So, it can connect in no time to MySQL, MongoDB, or any other database.
5. **Supportive Community:** PHP has a very supportive online community. The official documentation provides guides on how to use the features and you can easily get your problem fixed while stuck.

Disadvantages of PHP:

1. **Limited debugging tools:** One common complaint most developers have about PHP is that it has limited debugging tools. It handles errors poorly, especially when compared to other scripting languages. This is mainly because the debugging tools that are needed to track and search for such errors don't work as well.
2. **Security issues**: Since PHP is open source, it is not secure. The ASCII text file is easily available, and anyone can look it up.
3. **Not suitable for giant web application development:** If one wants to develop a giant content-based web application, it is not possible to do it with the help of PHP. They would have to use other programming languages.
4. **Extra learning:** One needs to know about the PHP framework to use the built-in PHP functionalities. If one does not know about it, they will be forced to write additional codes.

Features of PHP:

1. **Simple:** It is very simple and easy to use, compare to other scripting language it is very simple and easy, this is widely used all over the world.
2. **Open Source:** Open source means you no need to pay for use php, you can free download and use.

3. **Case Sensitive:** PHP is case sensitive scripting language at time of variable declaration. In PHP, all keywords (like if, else, while, echo, etc.), classes, functions and user-defined functions are case sensitive.
4. **Excellent database connectivity:** PHP most likely can connect to any database that exists. For instance- Access, Firebird, MySQL, Oracle, and iBase are among the most popular ones. One of the most crucial features and the biggest advantages of PHP language is its interoperability and cross-platform capabilities.
5. **Performance:** PHP uses its own memory, so its processing speed is fast as compared to other scripting language such as JSP and ASP.
6. **Embedded:** PHP code can be easily embedded with HTML and other script.

Data Types in PHP

The values assigned to a PHP variable may be of different data types including simple string and numeric types to more complex data types like arrays and objects. PHP supports total eight primitive data types: **Integer, Floating point number or Float, String, Booleans, Array, Object, resource and NULL**. These data types are used to construct variables. Examples are given below.

1. **PHP Integer:**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>PHP Integers</title>
</head>
<body>
     <h2>PHP Integers</h2>
     Integers are whole numbers, without a decimal
     point (..., -2, -1, 0, 1, 2, ...).<br><br>
     <?php
          $pn = 123; // posative number
          var_dump($pn);
          echo "<br>";
          echo "using echo <br>";
          echo ($pn);
          echo "<br>";
          $nn = -123; // a negative number
          var_dump($nn);
          echo "<br>";
          echo "using echo <br>";
          echo ($nn);
     ?>
</body>
</html>
```

2. **PHP Floating Point Numbers or Doubles**

```
<html lang="en">
<head>
    <title>PHP Floats</title>
</head>
<body>
     <h2>PHP Floating Point Numbers or Doubles</h2>
     Floating point numbers (also known as "floats",
     "doubles", or "real numbers") are decimal or fractional
     numbers.
     <br><br>
     <?php
          $a = 1.234;
          var_dump($a);
          echo "<br>";
          $b = 10.2e3;
          var_dump($b);
          echo "<br>";
          $c = 4E-10;
          var_dump($c);
     ?>
</body>
</html>
```

3. **PHP Array:**

```
<html>
<head>
     <title>PHP Arrays</title>
</head>
<body>
     <h2>PHP Arrays</h2>
     An array is a variable that can hold more than one value
     at a time. It is useful to aggregate a series of related
     items together, for example a set of country or city
     names.
     An array is formally defined as an indexed collection
     of data values. Each index (also known as the key) of
     an array is unique and references a corresponding
     value. <hr><hr>
     <b>Default Index value:</b><br>
     <?php
          $cars = array("Volvo","BMW","Toyota");
          var_dump($cars);
          ?>    <hr><hr>
          <b>User Defined Index value:</b><br>
          <?php
          $color_codes = array (
              "Red" => "#ff0000",
              "Green" => "#00ff00",
              "Blue" => "#0000ff"
          );
          var_dump($color_codes);
     ?>
</body>
</html>
```

4. PHP Strings

```
<html lang="en">
<head>
    <title>PHP Strings</title>
</head>
<body>
     <h2>PHP Strings</h2>
     Strings are sequences of characters, where every
     character is the same as a byte. A string can hold
     letters, numbers, and special characters and it can be
     as large as up to 2GB (2147483647 bytes maximum). The
     simplest way to specify a string is to enclose it in
     single quotes (e.g. 'Hello world!'), however you can also
     use double quotes ("Hello world!").<br><br>
     <?php
          $a = 'Hello world!';
          echo $a;
          echo "<br> ********** <br>";
          var_dump($a);
          echo "<br> ******** <br>";
          $b = "Hello world!";
          echo $b;
          echo "<br>";
          $c = 'Stay here, I\'ll be back.';
          echo $c;
     ?>
</body>
</html>
```

5. PHP Booleans:

```
<html lang="en">
<head>
    <title>PHP Booleans</title>
</head>
<body>
     <h2>PHP Booleans</h2>
     Booleans are like a switch it has only two possible
     values either 1 (true) or 0 (false).
     <br><br>
     <hr> Assign the value TRUE to a variable<br>
     <?php
          $show_error = True;
          var_dump($show_error);
          ?>
          <hr> Assign the value FALSE to a variable<br>
          <?php
          $show_error = False;
          var_dump($show_error);
     ?>
</body>
</html>
```

6. **PHP NULL:**

```
<html lang="en">
<head>
    <title>PHP NULL Value</title>
</head>
<body>
     <h2> PHP NULL</h2>
     The special NULL value is used to represent empty
     variables in PHP. A variable of type NULL is a
     variable without any data. NULL is the only possible
     value of type null.
     <br><br>
     <?php
          $a = NULL;
          var_dump($a);
          echo "<br>";
          $b = "Hello World!";
          var_dump($b);
          echo "<br>";
          $b = NULL;
          var_dump($b);
     ?>
</body>
</html>
```

PHP Operators:

1. Arithmetic Operators
2. Comparison Operators
3. Logical Operators
4. Assignment Operators

Operators	Purpose	Example	Same as
=	Assign Valeu to Right	C=a+b	C=a+b
+=	Add and Assignment	a+=b	a=a+b
-=	Subtract and Assignment	a-=b	a=a-b
*=	Multiply and Assignment	a*=b	a=a*b
/=	Divide and Assignment	a/=b	a=a/b
%=	Takes Modulus and assignment (Only for Integer)	a%=b	a=a%b

5. **Conditional Operators**

Operators	Purpose	Syntax	Example
?:	Conditional Expression	If condition is true? Then value X: Otherwise, Value Y	`<?php $age = 20; print ($age >= 18)? "Adult": "Not Adult"; ?>`

Super-global Variables:

Super-global Variables in PHP are predefined global variables. Global variables are variables with global scope, which means that they can be used wherever needed – they do not need to be declared, nor do they

need to be marked with global in functions. All super-global variables are written in all-caps like, $_GLOBALS. All the super-global variables act as associative arrays that use a string value as a key to access values. The following is a list of super-global variables in PHP:

1. **$_GLOBALS** is the super-global variable that stores all user-defined global variables of current page. Global variable names act as keys to their values.
2. **$_SERVER** contains data about headers, scripts, and paths. The keys to the values in this array are predefined.
3. **$_REQUEST** stores data input in the form of **HTTP, POST, GET** and **Cookies**. The keys to this array are defined in the HTTP requests.
4. **$_POST** stores data input in the form of POST requests. The keys to this array are defined in the HTTP POST request.
5. **$_GET** has data input in the form of GET requests. The keys to this array are defined in the HTTP GET request.
6. **$_FILES** is a two-dimensional associative array that contains a list of files that were uploaded to the script using the POST method. The keys to this array are the names of the fields uploading the files and the data being accessed. For example, `$_FILES[fileUploaded][FileName]` accesses the name of the file being uploaded from the *file Upload field*.
7. **$_COOKIES** keeps data input via HTTP Cookies. The keys to this array are defined when the cookies are set.
8. **$_SESSION** holds session variables. Session variables can be accessed on multiple pages. This array's keys are defined by the users when they define session variables.
9. **$_ENV** contains information about the environment that PHP is running in. The keys to the values in this array are predefined.

PHP Form Handling:

We can create and use forms in PHP. To get data from form, we need to use PHP super-global **$_GET** for get methos form and **$_POST** for post method form.

Get Method:

GET method sends the encoded user information appended to the page request. The page and the encoded information are separated by "?" character. The way of converting information into a header called QUERY_STRING.

```
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 sending up to 1024 characters only.
- Never use the 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.

The POST Method

The POST method transfers information via HTTP headers. The information is not converted to URL as described in the case of GET method.

- 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.

Database Connectivity:

Since PHP 5.5, mysql_connect() is deprecated (बहिष्कृत). Now it is recommended to use one of the following two alternatives.

1. Mysqli_connect() → Recommended
2. PDO::__construct()

PHP mysqli_connect():

PHP mysqli_connect() is used to connect with MySQL database. It returns resources if the connection is established or null. ***Syntax:*** *resource mysqli_connect(server, username,password);*

PHP mysqli_close():

PHP mysqli_close() function is used to disconnect with MySQL database. It returns true if connection is collected else return false. ***Syntax:*** *mysqli_close (resource $resource_link);*

Example:

```php
<?php
    $host="locallhost:3306";
    $user="root";
    $password="";
    $dbName="DataBaseName";
    $connect=mysqli_connect($host,$user,$password,$dbName);
    if(!connect){
        die('Could not connect :'.mysqli_error());
    }
    echo 'Connected Successfully';
    mysqli_close($connect);
?>
```

Structured Query Language:

Structured Query Language (SQL) is the language that we use to work with these relational database management systems. SQL(Originally called sequel) is a language that communicates with databases. Most relational databases, including MySQL, ORACLE, SQL Server, etc., support (SQL) query language.

Note: SQL is not case-sensitive.

Some Features of or Function of SQL

- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database
- SQL can create views in a database
- SQL can set permissions on tables, procedures, and views
- SQL can execute queries against a database