

# Differentiate between PHP5 and PHP7.

## 1. Performance:

- **PHP 5:** PHP 5 had performance limitations and was relatively slower compared to later versions.
- **PHP 7:** PHP 7 introduced significant performance improvements. It included the Zend Engine 3.0, which made PHP 7 up to twice as fast as PHP 5.6 for many applications. This improvement was achieved through better memory usage and optimizations in the core engine.

## 2. Scalar Type Declarations:

- **PHP 5:** PHP 5 allowed type hinting for class/interface types, but it didn't support scalar type declarations (int, float, string, bool).
- **PHP 7:** PHP 7 introduced scalar type declarations, which allow you to specify the expected data types for function parameters and return values. You can use `int`, `float`, `string`, and `bool` as type hints.

## 3. Return Type Declarations:

- **PHP 5:** PHP 5 did not support return type declarations for functions.
- **PHP 7:** PHP 7 introduced return type declarations, which allow you to specify the expected data type that a function should return. You can use : `returnType` after the function declaration.

## 4. Spaceship Operator (<=>):

- **PHP 5:** The spaceship operator (<=>) was not available in PHP 5.
- **PHP 7:** PHP 7 introduced the spaceship operator for comparing two values, making it easier to implement custom sorting and comparison functions.

## 5. Null Coalescing Operator (??):

- **PHP 5:** The null coalescing operator (??) was not available in PHP 5.
- **PHP 7:** PHP 7 introduced the null coalescing operator, which provides a concise way to handle potentially null values.

## 6. Anonymous Classes:

- **PHP 5:** PHP 5 did not support anonymous classes.
- **PHP 7:** PHP 7 introduced support for anonymous classes, allowing you to define classes on the fly without explicitly naming them.

## 7. Error Handling:

- **PHP 5:** PHP 5 used traditional error handling with `mysql_error()`, `mysql_errno()`, and `die()`.
- **PHP 7:** PHP 7 introduced a more consistent and improved error handling mechanism using exceptions. It also introduced the Throwable interface for better exception handling.

## 8. Improved Type Hinting:

- **PHP 5:** PHP 5 had limited support for type hinting, primarily for class and interface types.
- **PHP 7:** PHP 7 expanded type hinting capabilities, including scalar type hints, return type hints, and improved support for custom types through interfaces and classes.

## 9. Deprecated Features:

- **PHP 5:** PHP 5 included many features and functions that were deprecated and removed in later versions.
- **PHP 7:** PHP 7 removed several deprecated features and improved the consistency of the language.

## 10. Unicode Support:

- **PHP 5:** PHP 5 had limited support for Unicode, making handling multibyte character encodings challenging.
- **PHP 7:** PHP 7 improved Unicode support and introduced functions for working with multibyte strings more effectively.

## 11. New Operators and Functions:

- **PHP 5:** PHP 5 introduced some operators and functions, but the additions were more significant in PHP 7.

## 12. Compatibility:

- **PHP 5:** PHP 5 had a substantial user base and many existing applications running on it.
- **PHP 7:** PHP 7 aimed to maintain backward compatibility with PHP 5 while introducing new features. However, some older code might require modifications to run on PHP 7.

# Types of Error in php

## 1. Parse Errors:

- Parse errors occur during the parsing phase of PHP script execution, often due to syntax errors in the code.
- These errors prevent the script from running and typically include messages like "syntax error, unexpected token."

## 2. Fatal Errors:

- Fatal errors are severe issues that cause the script to terminate abruptly.
- Examples include calling an undefined function or accessing an undefined class.
- Fatal errors cannot be caught or handled within the script.

## 3. Warnings:

- Warnings are non-fatal errors that don't halt script execution but indicate potential issues.

- For example, using an uninitialized variable or including a file that doesn't exist.

#### 4. Notices:

- Notices are less severe than warnings and are often related to variables that are used **without being explicitly initialized**.
- While notices don't halt script execution, they should be addressed to ensure correct behavior.

#### 5. Errors Triggered by User-defined Functions:

- These errors can occur within user-defined functions or methods.
- Developers can use the `trigger_error()` function to generate custom errors and exceptions within their code.

#### 6. Exceptions:

- Exceptions are a more advanced error-handling mechanism introduced in PHP 5.
- They allow for structured error handling using `try`, `catch`, `finally`, and `throw` statements.
- Exceptions can be categorized into two main types:
  - **Built-in Exceptions:** PHP provides several built-in exception classes, such as `Exception`, `InvalidArgumentException`, `RuntimeException`, etc.
  - **Custom Exceptions:** Developers can define their own custom exception classes to handle specific application-level errors.

#### 7. Deprecated Features:

- Deprecated features are functions, classes, or methods that are marked as deprecated and may be removed in future PHP versions.
- Using deprecated features generates warnings to encourage developers to update their code.

#### 8. Compile-time Errors:

- Compile-time errors occur during the compilation phase before script execution.
- These errors include issues like referencing a **non-existent class** or extending a final class.

#### 9. Execution-time Errors:

- Execution-time errors occur during script execution and may be triggered by various runtime conditions.
- Examples include division by zero, accessing an undefined array index, or exceeding memory limits.

#### 10. Assertion Errors:

- Assertions are used to test conditions during development and are typically disabled in production.
- When an assertion fails, an `AssertionError` is thrown, indicating a logical error in the code.

#### 11. Database Errors:

- These errors occur when interacting with a database using database extensions like MySQLi or PDO.
- Examples include connection errors, query errors, and data retrieval errors.

## 12. HTTP Errors:

- HTTP errors are not PHP-specific but may be relevant when dealing with web applications.
- Examples include HTTP status codes like 404 (Not Found) or 500 (Internal Server Error).

# LAMP Setup

Setting up a LAMP (Linux, Apache, MySQL, PHP) stack on a server involves installing and configuring the necessary components to host web applications or websites. Here are the steps to set up a LAMP stack on a Linux server:

### 1. Provision a Linux Server:

1. Choose a Linux distribution such as Ubuntu, CentOS, or Debian for your server.
2. Rent a virtual private server (VPS) from a hosting provider or set up a physical server, if applicable.

### 2. Connect to the Server:

Access your server via SSH using a terminal or an SSH client. Replace `your_server_ip` with the actual IP address of your server.

css

Copy code

```
ssh username@your_server_ip
```

### 3. Update the System:

Before installing any software, update the package list and upgrade installed packages to ensure you have the latest security updates.

sql

Copy code

```
sudo apt update sudo apt upgrade
```

### 4. Install Apache Web Server:

Install the Apache web server using the package manager for your Linux distribution. For example, on Ubuntu:

Copy code  
sudo apt install apache2

## 5. Start and Enable Apache:

Start the Apache service and enable it to start automatically on boot.

Copy code  
bash  
sudo systemctl start apache2 sudo systemctl enable apache2

## 6. Test Apache:

Open a web browser and enter your server's IP address. You should see the default Apache test page, indicating that Apache is running.

## 7. Install MySQL (or MariaDB):

Install the MySQL or MariaDB database server. MariaDB is a drop-in replacement for MySQL and is often used in modern LAMP setups.

Copy code  
sudo apt install mariadb-server

## 8. Secure MySQL (Optional but Highly Recommended):

Run the MySQL secure installation script to set a root password and improve security.

Copy code  
sudo mysql\_secure\_installation

## 9. Install PHP:

Install PHP and necessary PHP modules for your web applications.

Copy code  
sudo apt install php php-mysql

## 10. Test PHP:

Create a PHP info file to test PHP installation. Create a file named `info.php` in the web server's document root (usually `/var/www/html`) with the following content:

Copy code  
php  
<?php phpinfo(); ?>

Access `http://your_server_ip/info.php` in a web browser to see PHP information.

## **11. Configure Apache for PHP:**

To enable Apache to process PHP files, you may need to configure Apache to use the PHP module. On Ubuntu, it's usually enabled by default. On other distributions, you might need to enable it manually.

## **12. Create and Host Your Web Applications:**

Upload your web application files to the appropriate directory (usually `/var/www/html`) and configure Apache virtual hosts if needed.

## **13. Secure Your Server:**

1. Implement firewall rules (e.g., using UFW or iptables) to restrict access to your server.
2. Keep software up to date, apply security patches, and follow best practices for server security.

## **14. Optional: Install phpMyAdmin (Database Management Tool):**

If you need a web-based interface for managing MySQL/MariaDB databases, you can install phpMyAdmin. Be sure to secure it properly.

That's it! You now have a basic LAMP stack set up on your server. You can expand upon this setup by installing additional PHP modules, configuring Apache virtual hosts, and securing your web applications as needed.

C:\xampp\htdocs\test\demo.php - Sublime Text (UNREGISTERED)

```
1 <?php
2 $db=new mysqli("localhost","root","","testdemo");
3 if(isset($_POST['submit'])){
4 $check=implode(' ', $_POST['ch']);
5 $country=$_POST['country'];
6 $qry=$db->query("INSERT INTO `test`(`checkbox`, `country`) VALUES ('$check','$country')");
7 if($qry>0){
8     echo "<script> alert('Data is Submitted');</script>";
9 }else{
10    echo "<script> alert('Data is not Submitted');</script>";
11 }
12 }
13 ?>
14 <!DOCTYPE html>
15 <html>
16 <head>
17     <title>Demo</title>
18 </head>
19 <body>
20 <form action="" method="POST">
21     <input type="checkbox" name="ch[]" value="PHP">PHP<br>
22     <input type="checkbox" name="ch[]" value="JAVA">JAVA<br>
23     <input type="checkbox" name="ch[]" value="CSS">CSS<br>
24     <input type="checkbox" name="ch[]" value="HTML">HTML<br>
25     <input type="checkbox" name="ch[]" value="CORE PHP">CORE PHP<br>
26     <select name="country">
```

Line 4, Column 35

Tab Size: 4

PHP

12:04 PM 2/4/2018

```
<?php
$con=mysqli_connect("localhost","root","","complete_php");
//$sql="insert into student(name,city) values('Vishal','Delhi')";
//$sql="update student set name='Amit' where id=1";
//$sql="delete from student where id=1";
$sql="select * from student";
$res=mysqli_query($con,$sql);
while($row=mysqli_fetch_assoc($res)){
    echo '<pre>';
    print_r($row);
}
?>
```

## Select Insert

Reference Cyber variye youtube chanel

[https://www.youtube.com/watch?v=UNFdrhu\\_5GU&list=PLEjKBGxF74J67G7KSGHA4JPOKW9H3cQq3&index=9](https://www.youtube.com/watch?v=UNFdrhu_5GU&list=PLEjKBGxF74J67G7KSGHA4JPOKW9H3cQq3&index=9)

```
<?php
    if($_POST['register'])
    {
        $fname = $_POST['fname'];
        $lname = $_POST['lname'];
        $pwd = $_POST['password'];
        $cpwd = $_POST['conpassword'];
        $gender = $_POST['gender'];
        $email = $_POST['email'];
        $phone = $_POST['phone'];
        $address = $_POST['address'];

        $query = "INSERT INTO FORM VALUES('$fname','$lname','$pwd','$cpwd','$gender','$email','$phone','$address')";
        $data = mysqli_query($conn,$query);

        if($data)
        {
            echo "Data Inserted into Database";
        }
        else
        {
            echo "Failed";
        }
    }
?>
```

```
<div class="input_field">
    <label>Gender</label>
    <div class="custom_select">
        <select name="gender">
            <option value="Not Selected">Select</option>
            <option value="male">Male</option>
            <option value="female">Female</option>
        </select>
    </div>
</div>
```

```
if($total != 0)
{
    while($result = mysqli_fetch_assoc($data))
    {
        echo $result[fname]." ".$result[lname]." ".$result[gender]." ".$
            result[email]." ".$result[phone]." ".$result[address]."<br>";
    }
}
else
{
    echo "No records found";
}
```

```
<?php
while($result = mysqli_fetch_assoc($data))
{
    echo "<tr>
        <td>".$result[fname]."</td>
        <td>".$result[ lname]."</td>
        <td>".$result[ gender]."</td>
        <td>".$result[ email]."</td>
        <td>".$result[ phone]."</td>
        <td>".$result[ address]."</td>
    </tr>
    ";
}
```

```
<div class="input_field">
    <label>Address</label>
    <textarea class="textarea" name="address" required>
        <?php echo $result['address']; ?>
    </textarea>
</div>
```

## Updated

```
<select name="gender" required>
    <option value="">Select</option>

    <option value="male"
        <?php
            if($result['gender'] == 'male')
            {
                echo "selected";
            }
        ?>
    >

        Male</option>
    <option value="female"
        <?php
            if($result['gender'] == 'female')
            {
                echo "selected";
            }
        ?>
    >

        Female</option>
</select>
```

```
$email = $_POST['email'];
$phone = $_POST['phone'];
$address = $_POST['address'];

$query = "UPDATE form2 set fname='$fname',lname='$lname',password
        ='$pwd',cpassword='$cpwd',gender='$gender',email='$email',
        phone='$phone',address='$address' WHERE id='$id'";

$data = mysqli_query($conn,$query);

if($data)
{
    echo "Record Updated";
}
else
{
    echo "Failed to Update";
}
```

```
$query = "UPDATE form2 set fname='$fname',lname='$lname',password='$
        pwd',cpassword='$cpwd',gender='$gender',email='$email',phone='$
        phone',address='$address' WHERE id='$id'";

$data = mysqli_query($conn,$query);

if($data)
{
    echo "<script>alert('Record Updated')</script>";
    ?>
    <meta http-equiv = "refresh" content = "0; url =
    http://localhost:8080/crud/display.php" />
    <?php
}
else
{
    echo "Failed to Update";
}
```

```
        <a href='delete.php?id=$result[id]'><input type='submit'  
            value='Delete' class='delete' onclick = 'return  
            checkdelete()'></a></td>  
    </tr>  
    "  
}  
  
<script>  
    function checkdelete()  
    {  
        return confirm('Are you sure your want to delete this record ?');  
    }  
</script>
```

```
$id = $_GET['id'];  
  
$query = "DELETE FROM FORM2 WHERE id = '$id' ";  
$data = mysqli_query($conn,$query);  
  
if($data)  
{  
    echo "<script>alert('Record Deleted')</script>";  
    ?>  
    <meta http-equiv = "refresh" content = "0; url = http://localhost:8080  
    /crud/display.php" />  
    <?php  
}  
else  
{  
    echo "<script>alert('Failed To Deleted')</script>";  
}  
?  
?
```

## Redio

Phone Number

Caste

General  OBC  SC  ST

Language

Hindi  Urdu  English

Address

```
<div class="input_field">
    <label>Phone Number</label>
    <input type="text" class="input" name="phone" required>
</div>

<div class="input_field">
    <label style="margin-right: 100px;">Caste</label>
    <input type="radio" name="r1" value="General" required><label style="margin-left: 5px;">General</label>
    <input type="radio" name="r1" value="OBC" required><label style="margin-left: 5px;">OBC</label>
    <input type="radio" name="r1" value="SC" required><label style="margin-left: 5px;">SC</label>
    <input type="radio" name="r1" value="ST" required><label style="margin-left: 5px;">ST</label>
</div>

<div class="input_field">
    <label>Address</label>
    <textarea class="textarea" name="address" required></textarea>
</div>
```

```
<?php
if($_POST['register'])
{
    $fname = $_POST['fname'];
    $lname = $_POST['lname'];
    $pwd = $_POST['password'];
    $cpwd = $_POST['cpassword'];
    $gender = $_POST['gender'];
    $email = $_POST['email'];
    $phone = $_POST['phone'];
    $caste = $_POST['r1'];
    $address = $_POST['address'];

    $query = "INSERT INTO FORM2 (fname, lname, password, cpassword, gender, email, phone, caste, address)
              VALUES ('$fname', '$lname', '$pwd', '$gender', '$email', '$phone', '$caste', '$address')";
    $data = mysqli_query($conn, $query);

    if($data)
    {
        echo "Data Inserted into Database";
    }
    else
    {
```

```

<div class="input_field">
    <label style="margin-right: 100px;">Caste</label>
    <input type="radio" name="r1" value="General" required

    <?php
        if($result[caste] == "General")
        {
            echo "checked";
        }
    ?>

    <label style="margin-left: 5px;">General</label>
    <input type="radio" name="r1" value="OBC" required

    <?php
        if($result[caste] == "OBC")
        {
            echo "checked";
        }
    ?>

```

Upload Image

Choose File
No file chosen

---

First Name

Last Name

---

Password

Confirm Password

---

Gender

---

Email Address

---

Phone Number

---

Caste

General    OBC    SC    ST

---

Language

Hindi    Urdu    English

---

Address

---

Agree to terms and conditions

## Checkbox

```
<div class="input_field">
    <label style="margin-right: 100px;">Language</label>
    <input type="checkbox" name="language[]" value="Hindi" required><label style="margin-left: 5px;">Hindi</label>
    <input type="checkbox" name="language[]" value="Urdu" required><label style="margin-left: 5px;">Urdu</label>
    <input type="checkbox" name="language[]" value="English" required><label style="margin-left: 5px;">English</label>
</div>

<div class="input_field">
    <label>Address</label>
    <textarea class="textarea" name="address" required></textarea>
</div>
```

```
{
    $fname = $_POST['fname'];
    $lname = $_POST['lname'];
    $pwd = $_POST['password'];
    $cpwd = $_POST['conpassword'];
    $gender = $_POST['gender'];
    $email = $_POST['email'];
    $phone = $_POST['phone'];
    $caste = $_POST['r1'];

    $lang = $_POST['language'];
    $lang1 = implode(",",$lang);

    $address = $_POST['address'];

    $query = "INSERT INTO FORM2 (fname, lname, password, cpassword, gender, email,
        phone, caste, language, address) VALUES('$fname', '$lname', '$pwd', '$cpwd', '$
        gender', '$email', '$phone', '$caste', '$lang1', '$address')";
    $data = mysqli_query($conn, $query);

    if($data)
    {
        echo "Data Inserted into Database";
    }
}
```

```
$id = $_GET['id'];

$query = "SELECT * FROM FORM4 where id= '$id'";
$data = mysqli_query($conn, $query);
$result = mysqli_fetch_assoc($data);

$language = $result['language'];
$language1 = explode(", ", $language)

?>

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

```

<label style="margin-right: 100px;">Language</label>
    <input type="checkbox" name="language[]" value="Hindi"
        <?php
        if(in_array(Hindi, $language1))
        {
            echo "checked";
        }
        ?>

    >

<label style="margin-left: 5px;">Hindi</label>
<input type="checkbox" name="language[]" value="Urdu"

<?php
if(in_array(Urdu, $language1))
{
    echo "checked";
}
?>

>

```

```

display.php      update_design.php      form.php
0   $phone      = $_POST['phone'];
1   $caste      = $_POST['r1'];
2
3   $lang       = $_POST['language'];
4   $lang1      = implode(",",$lang);
5
6   $address    = $_POST['address'];
7
8   $query = "UPDATE form4 set fname='$fname',lname='$lname',password='$pwd'
              ,cpassword='$cpwd',gender='$gender',email='$email',phone='$phone',
              caste='$caste',language='$lang1',address='$address' WHERE id='$id'";
9
10  $data = mysqli_query($conn,$query);
11
12  if($data)
13  {
14      echo "<script>alert('Record Updated')</script>";
15      ?>
16
17      <meta http-equiv = "refresh" content = "0; url =
18          http://localhost:8080/crud/display.php" />
19
20      </?php

```

### After Insert data Redirection

```
$address = $_POST['address'];

$query = "INSERT INTO FORM4 (std_image,fname, lname, password, cpassword, gender,
    email, phone, caste, language, address) VALUES('$folder', '$fname', '$lname', '$pwd',
    '$cpwd', '$gender', '$email', '$phone', '$caste', '$lang1', '$address')";
$data = mysqli_query($conn,$query);

if($data)
{
    echo "<script> alert('Data Inserted into Database') </script>";
}
else
{
    echo "<script> alert('Failed') </script>";
}

?>
```

```
<body>
<?php
    $connection = mysqli_connect("localhost", "root", "", "aptech");
    $query = "SELECT * FROM tbl_products";
    $result = mysqli_query($connection, $query);
?>
<select>
    <option>Select Any Product</option>
    <?php
        foreach($result as $row)
        {
            echo "<option>$row[product_name]</option>";
        }
    ?>
</select>
</body>
```

## File Upload

```
<body>
    <form action="#" method="POST" enctype="multipart/form-data">
        <input type="file" name="uploadfile"><br><br>
        <input type="submit" name="submit" value="Upload File">
    </form>
</body>
```

```
<?php

//print_r($_FILES["uploadfile"]);
$filename = $_FILES["uploadfile"]["name"];
$tempname = $_FILES["uploadfile"]["tmp_name"];
$folder = "images/".$filename;
echo $folder;
move_uploaded_file($tempname, $folder);

?>
```

```
<?php

//print_r($_FILES["uploadfile"]);
$filename = $_FILES["uploadfile"]["name"];
$tempname = $_FILES["uploadfile"]["tmp_name"];
$folder = "images/".$filename;
//echo $folder;
move_uploaded_file($tempname, $folder);
    I
echo "<img src='$folder' height='100px' width='100px' ";

?>

```

```
<?php  
if(isset($_POST['submit'])) {  
    echo '<pre>';  
    print_r($_FILES);  
    move_uploaded_file($_FILES['doc']['tmp_name'],  
        'media/'.$_FILES['doc']['name']);  
}  
  
?>  
<form method="post" enctype="multipart/form-data">  
    <input type="file" name="doc"/>  
    <input type="submit" name="submit"/>  
</form>
```

```
1  <?php  
2  if(isset($_POST['submit'])) {  
3      $file=rand(111111111,999999999);  
4  
5      echo '<pre>';  
6      print_r($_FILES);  
7      move_uploaded_file($_FILES['doc']['tmp_name'],  
8          'media/'.$file.'_'.$_FILES['doc']['name']);  
9  
10 ?>  
11 <form method="post" enctype="multipart/form-data">  
12     <input type="file" name="doc"/>  
13     <input type="submit" name="submit"/>  
14 </form>
```

```
fileupload.php * form.php * display.php
101 </body>
102
103 </html>
104
105
106 <?php
107     if($_POST['register'])
108     {
109
110
111     $filename = $_FILES["uploadfile"]["name"];
112     $tmpname = $_FILES["uploadfile"]["tmp_name"];
113     $folder = "images/".$filename;
114     move_uploaded_file($tmpname, $folder);
115
116
117     $fname    = $_POST['fname'];
118     $lname   = $_POST['lname'];
119     $pwd      = $_POST['password'];
120     $cpwd    = $_POST['conpassword'];
121     $gender   = $_POST['gender'];
122     $email    = $_POST['email'];
123     $phone    = $_POST['phone'];

$lang    = $_POST['language'];
$lang1   = implode(",",$lang);

$address = $_POST['address'];

$query = "INSERT INTO FORM4 (std_image, fname, lname, password, cpassword, gender,
    email, phone, caste, language, address) VALUES( '$folder|', '$fname', '$lname', '$
    pwd', '$cpwd', '$gender', '$email', '$phone', '$caste', '$lang1', '$address')";
$data = mysqli_query($conn,$query);

if($data)
{
    echo "Data Inserted into Database";
}
else
{
```

```

<?php
while($result = mysqli_fetch_assoc($data))
{
    echo "<tr>
        <td>".$result[id]."</td>

        <td><img src= '".$result[std_image]."' height='100px' width='100px'></td>

        <td>".$result[fname]."</td>
        <td>".$result[lname]."</td>
        <td>".$result[gender]."</td>
        <td>".$result[email]."</td>
        <td>".$result[phone]."</td>
        <td>".$result[caste]."</td>
        <td>".$result[language]."</td>
        <td>".$result[address]."</td>

        <td><a href='update_design.php?id=$result[id]'\><input type='submit' value='Update' class='update'\></a>

        <a href='delete.php?id=$result[id]'\><input type='submit' value='Delete' class='delete' onclick = 'return checkdelete()'\></a></td>
    </tr>

```

## PHP file upload

```

<?php
if(isset($_POST['submit'])){
    //move_uploaded_file();
    foreach($_FILES['doc']['name'] as $key=>$val){
        $rand=rand('11111111','99999999');
        $file=$rand.'_'.$val;
        move_uploaded_file($_FILES['doc']['tmp_name'][$key], 'media/'.$file);
        //insert into table(image) values('$file');
    }
}
?>
<form method="post" enctype="multipart/form-data">
    <input type="file" name="doc[]" multiple/>
    <input type="submit" name="submit"/>
</form>

```

## PHP oops file upload

*upload.php*

```
<?php
class upload{
    public $src = "./upload/";
    public $tmp;
    public $filename;
    public $type;
    public $uploadfile;

    public function startupload(){
        $this -> filename = $_FILES["file"]["name"];
        $this -> tmp = $_FILES["file"]["tmp_name"];
        $this -> uploadfile = $src . basename($this -> name);
    }
    public function uploadfile(){
        if(move_uploaded_file($this -> tmp, $this ->
uploadFile)){
            return true;
        }
    }
}

?>
```

*index.php*

```
<?php
require_once('./lib/upload.php');
?>
<?php
if(isset($_POST['file'])){
    $fileupload = new upload();
    if($fileupload -> uploadfile()){
        echo 'Success';
    }
}
?>

<html>
```

```

<head></head>
<body>
<form align="center" enctype="multipart/form-data" action="<?php
echo htmlspecialchars($_SERVER["PHP_SELF"]); ?>" method="post">
Select upload file: <input type="file" name="file"
required="yes" />
<input type="submit" value="Submit" />
<p>
</form>
</body>
</html>

```

## Upload Multiple file

<https://www.studentstutorial.com/php/file-upload-in-php-mysql>

```

<!DOCTYPE html>
<html>
<head>
<title>File Upload</title>
</head>
<body>
<form action="upload.php" method="post" enctype="multipart/form-
data">
<input type="file" name="file" />
<button type="submit" name="upload">upload</button>
</form>
</body>
</html>

```

### upload.php

```

<?php
include_once 'database.php';
if(isset($_POST['upload']))
{
    $file = rand(1000,100000)."-".$_FILES['file']['name'];
    $file_loc = $_FILES['file']['tmp_name'];
    $file_size = $_FILES['file']['size'];
    $file_type = $_FILES['file']['type'];
    $folder="upload/";

    /* new file size in KB */
    $new_size = $file_size/1024;
}

```

```

/* new file size in KB */

/* make file name in lower case */
$new_file_name = strtolower($file);
/* make file name in lower case */

$final_file=str_replace(' ','-',$new_file_name);

if(move_uploaded_file($file_loc,$folder.$final_file))
{
    $sql="INSERT INTO image(file,type,size)
VALUES('$final_file','$file_type','$new_size')";
mysqli_query($conn,$sql);

echo "File sucessfully upload";

}

else
{

echo "Error.Please try again";

}

?>

```

## For Multiple File

```

<!DOCTYPE html>
<html lang="en" >
<head>
<meta charset="UTF-8">
<title>Multiple image upload</title>
</head>
<body>
<form method="post" action="process.php">
<input type="file" name="image[]" multiple="multiple" >
<p align="center"><button type="submit" class="btn btn-warning"
id="butsave">Submit<span class="glyphicon glyphicon-
send"></span></button></p>
</form>
</body>
</html>

```

## *process.php*

```
<?php
$output_dir = "upload/"; /* Path for file upload */
$fileCount = count($_FILES["image"]['name']);
for($i=0; $i < $fileCount; $i++)

{
$RandomNum    = time();

$imageName     = str_replace(' ', '-',
',strtolower($_FILES['image']['name'][$i]));
$imageType     = $_FILES['image']['type'][$i];
/*"image/png", image/jpeg etc.*'

$imageExt = substr($imageName, strpos($imageName,
'.'));
$imageExt     = str_replace('.','',$imageExt);
$imageName     = preg_replace("/\.\[^.\s]{3,4}\$/",
'', $imageName);
$newImageName = $imageName.'-
' . $RandomNum . '.' . $imageExt;

$ret[$newImageName] = $output_dir.$newImageName;

/* Try to create the directory if it does not exist
*/
if (!file_exists($output_dir . $last_id))
{
    @mkdir($output_dir . $last_id, 0777);
}

move_uploaded_file($_FILES["image"]["tmp_name"][$i],$output_dir.
$last_id."/". $newImageName);

/*$insert_img = "insert into `category_images` SET
`category_ads_id`='".$category_ads_id_image."',
`image`='".$newImageName."'";
$result = $dbobj->query($insert_img);*/
}

echo "Image Uploaded Successfully";
?>
```

## Pegination

```
<?php

// Sample data array (replace this with your actual data source)
$data = range(1, 100); // Array with 1 to 100 elements

// Number of items per page
$itemsPerPage = 10;

// Get current page number from the URL, default to 1
$page = isset($_GET['page']) ? intval($_GET['page']) : 1;

// Calculate the offset for the data array
$offset = ($page - 1) * $itemsPerPage;

// Get the current page's data
$pageData = array_slice($data, $offset, $itemsPerPage);

// Display the data
echo "<h1>Page $page</h1>";
echo "<ul>";
foreach ($pageData as $item) {
    echo "<li>$item</li>";
}
echo "</ul>";

// Pagination links
$totalPages = ceil(count($data) / $itemsPerPage);

echo "<div>";
for ($i = 1; $i <= $totalPages; $i++) {
    echo "<a href='?page=$i'>$i</a> ";
}
echo "</div>";
?>
```

## Pagination Vishal

```
<?php
$con=mysqli_connect('localhost','root','','youtube');

$per_page=5;
$start=0;

if(isset($_GET['start'])){
    $start=$_GET['start'];
    $start--;
    $start=$start*$per_page;
}
$record=mysqli_num_rows(mysqli_query($con,"select id,title from page"));
$pagi=ceil($record/$per_page);

$sql="select id,title from page limit $start,$per_page";
$res=mysqli_query($con,$sql);
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Pagination Example</title>

<div class="container mt-100">
    <h2 class="mb-30">Pagination Example</h2>
    <ul class="list-group">
        <?php while($row=mysqli_fetch_assoc($res)){?>
            <li class="list-group-item"><?php echo $row['title']?></li>
        <?php } ?>
    </ul>
    <ul class="pagination mt-30">
        <?php for($i=1;$i<=$pagi;$i++){?>
            <li class="page-item"><a class="page-link" href="?start=<?php echo $i?><?php echo $i?></a></li>
        <?php } ?>
    </ul>
</div>
```

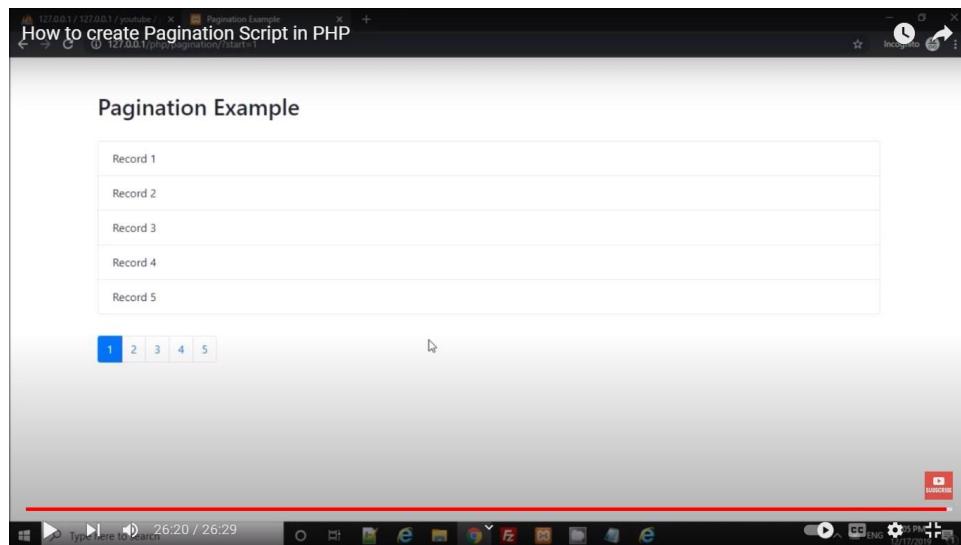
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Pagination Example</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <script src="js/jquery.min.js"></script>
    <script src="js/bootstrap.min.js"></script>
    <style>
        .mt-100{margin-top:50px;} .mt-30{margin-top:30px;} .mb-30{margin-bottom:30px;}
    </style>
<?php
$con=mysqli_connect ('localhost','root','','voutube');
$pri_page=5;
$start=0;
$current_page=1;
if(isset($_GET['start'])){
    $start=$_GET['start'];
    if($start<=0){
        $start=0;
        $current_page=1;
    }else{
        $current_page=$start;
        $start--;
        $start=$start*$pri_page;
    }
}
$record=mysqli_num_rows(mysqli_query($con,"select id,title from page"));
$pagi=ceil($record/$pri_page);

$sql="select id,title from page limit $start,$pri_page";
$res=mysqli_query($con,$sql);
?>
```

```

</head> <body>
<div class="container mt-100">
    <h2 class="mb-30">Pagination Example</h2>
    <ul class="list-group">
        <?php
        if(mysqli_num_rows($res)>0) {
            while($row=mysqli_fetch_assoc($res)) {?>
                <li class="list-group-item"><?php echo $row['title']?></li>
            <?php } } else {?>
                No records
            <?php } ?>
        </ul>
        <ul class="pagination mt-30">
            <?php
            for($i=1;$i<=$pagi;$i++) {
                $class='';
                if($current_page==$i) {
                    ?><li class="page-item active"><a class="page-link" href="javascript:void(0)">
                        <?php echo $i?></a></li>
                <?php
                }else{
                    ?>
                    <li class="page-item"><a class="page-link" href="?start=<?php echo $i?>">
                        <?php echo $i?></a></li>
                <?php
                }
                ?>
            <?php } ?>
        </ul>
    </div> </body> </html>

```



<https://www.youtube.com/watch?v=t0gBYc7QEU8>

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Pagination Example</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <script src="js/jquery.min.js"></script>
    <script src="js/bootstrap.min.js"></script>
    <style>
        .mt-100{margin-top:50px;} .mt-30{margin-top:30px;}.mb-30{margin-bottom:30px;}
    </style>
    <?php
$con=mysqli_connect('localhost','root','','youtube');

$per_page=5;
$start=0;
$current_page=1;
if(isset($_GET['start'])){
    $start=$_GET['start'];
    if($start<=0){
        $start=0;
        $current_page=1;
    }else{
        $current_page=$start;
        $start--;
        $start=$start*$per_page;
    }
}
$record=mysqli_num_rows(mysqli_query($con,"select id,title from page"));
$pagi=ceil($record/$per_page);

$sql="select id,title from page limit $start,$per_page";
$res=mysqli_query($con,$sql);
?>
</head>
<body>
<div class="container mt-100">
    <h2 class="mb-30">Pagination Example</h2>
    <ul class="list-group">
        <?php
        if(mysqli_num_rows($res)>0){
            while($row=mysqli_fetch_assoc($res)){?>
```

```

        <li class="list-group-item"><?php echo $row['title']?></li>
<?php } } else {?>
No records
<?php } ?>
</ul>
<ul class="pagination mt-30">
    <?php
    for($i=1;$i<=$pagi;$i++){
    $class=' ';
    if($current_page==$i){
        ?><li class="page-item active"><a class="page-link"
href="javascript:void(0)"><?php echo $i?></a></li><?php
    }else{
        ?>
            <li class="page-item"><a class="page-link" href="?start=<?php echo
$i?>"><?php echo $i?></a></li>
            <?php
        }
        ?>
    <?php } ?>
</ul>
</div>

</body>
</html>

```

```

1 | CREATE TABLE `tbluser` (
2 | `id` int(11) NOT NULL,
3 | `Name` varchar(120) NOT NULL,
4 | `PhoneNumber` int(11) NOT NULL,
5 | `Emailid` varchar(150) NOT NULL,
6 | `PostingDate` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP
7 | ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

```

1 <?php
2 // Database Configuration file
3 include('config.php');?>
4 <html>
5 <head>
6   <title>Pagination</title>
7   <!-- Bootstrap CDN -->
8   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
9   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
10  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
11 </head>
12 <body>
13 <table class="table">
14   <tr>
15     <th>#</th>
16     <th>Name</th>
17     <th>Phone Number</th>
18     <th>Email</th>
19     <th>Date</th>
20   </tr>
21 <?php
22 //Getting default page number
23   if (isset($_GET['pageno'])) {
24     $pageno = $_GET['pageno'];
25   } else {
26     $pageno = 1;
27   }
28 // Formula for pagination
29   $no_of_records_per_page = 10;
30   $offset = ($pageno-1) * $no_of_records_per_page;
31 // Getting total number of pages
32   $total_pages_sql = "SELECT COUNT(*) FROM tbluser";
33   $result = mysqli_query($con,$total_pages_sql);
34   $total_rows = mysqli_fetch_array($result)[0];
35   $total_pages = ceil($total_rows / $no_of_records_per_page);
36   $sql = "SELECT * FROM tbluser LIMIT $offset, $no_of_records_per_page";
37   $res_data = mysqli_query($con,$sql);
38   $cnt=1;
39   while($row = mysqli_fetch_array($res_data)){?>
40     <tr>
41       <td><?php echo $cnt;?></td>
42       <td><?php echo $row['Name'];?></td>
43       <td><?php echo $row['PhoneNumber'];?></td>
44       <td><?php echo $row['Emailid'];?></td>
45       <td><?php echo $row['PostingDate'];?></td>
46     </tr>
47   <?php
48   $cnt++;
49   ?>
50 </table>
51 <div align="center">
52   <ul class="pagination" >
53     <li><a href="?pageno=1">First</a></li>
54     <li class="disabled"><?php if($pageno <= 1){ echo 'disabled'; } ?><a href="<?php if($pageno <= 1){ echo '#'; } else { echo "?pageno=".($pageno - 1); } ?>">Prev</a>
55     </li>
56     <li class="disabled"><?php if($pageno >= $total_pages){ echo 'disabled'; } ?><a href="<?php if($pageno >= $total_pages){ echo '#'; } else { echo "?pageno=".($pageno + 1); } ?>">Next</a>
57     </li>
58     <li><a href="?pageno=<?php echo $total_pages; ?>">Last</a></li>
59   </ul>
</div>

```

# Paypal Integration in PHP

## Step 1: Create a PayPal Business Account

If you don't already have a PayPal business account, you need to create one. This account will allow you to receive payments.

## Step 2: Get PayPal API Credentials

PayPal API Username

PayPal API Password

PayPal API Signature

## Step 3: Set Up Your PHP Application

In your PHP application, create a form to collect payment information from users. Here's a simplified example of a payment form:

```
<!DOCTYPE html>
<html>
<head>
    <title>Pay with PayPal</title>
</head>
<body>
    <h2>Pay with PayPal</h2>
    <form action="process_payment.php" method="post">
        <input type="hidden" name="cmd" value="_xclick">
        <input type="hidden" name="business" value="your-paypal-
email@example.com">
        <input type="hidden" name="item_name" value="Product Name">
        <input type="hidden" name="item_number" value="123">
        <input type="hidden" name="amount" value="10.00">
        <input type="hidden" name="currency_code" value="USD">
        <input type="hidden" name="return" value="http://your-
website.com/thank-you.php">
        <input type="submit" name="submit" value="Pay with PayPal">
    </form>
</body>
</html>
```

#### Step 4: Create a PHP Script to Process Payments (process\_payment.php)

```
<?php
$paypal_url = 'https://www.sandbox.paypal.com/cgi-bin/webscr'; // Use the
sandbox URL for testing

// PayPal API credentials
$paypal_email = 'your-paypal-email@example.com'; // Your PayPal business
email
$paypal_currency = 'USD';

// Collect payment data from the form
$item_name = $_POST['item_name'];
$item_number = $_POST['item_number'];
$amount = $_POST['amount'];

// Build PayPal request parameters
$fields = [
    'cmd' => '_xclick',
    'business' => $paypal_email,
    'item_name' => $item_name,
    'item_number' => $item_number,
    'amount' => $amount,
    'currency_code' => $paypal_currency,
    'return' => 'http://your-website.com/thank-you.php', // Redirect URL
after payment
];
// Prepare the query string
$query_string = http_build_query($fields);

// Redirect to PayPal to complete the payment
header("location:$paypal_url?$query_string");
exit();
?>
```

#### Step 5: Handle the PayPal Response

#### Step 6: Test in the PayPal Sandbox

## PHP program with a trait and a class that uses that trait:

```
<?php
// Define a trait called Logger
trait Logger {
    public function log($message) {
        echo "Logging: $message\n";
    }
}

// Create a class that uses the Logger trait
class MyClass {
    use Logger;

    public function someMethod() {
        $this->log("This is a log message from MyClass");
    }
}

// Create another class that uses the Logger trait
class AnotherClass {
    use Logger;

    public function anotherMethod() {
        $this->log("This is a log message from AnotherClass");
    }
}

// Create instances of MyClass and AnotherClass
$obj1 = new MyClass();
$obj2 = new AnotherClass();

// Call methods that use the log method from the trait
$obj1->someMethod();
$obj2->anotherMethod();
?>
```

```

1 <?php
2 class class1{
3     function __construct(){
4         echo "start";
5     }
6
7     function fun1(){
8         echo "Hello";
9     }
10
11    function __destruct(){
12        echo "end";
13    }
14 }
15
16 $obj1=new class1();
17 $obj1->fun1();
18 ?>

```

```

<?php
class class1{
    function fun1(){
        echo "Hello";
    }
}

function __construct(){
    echo "start";
}

function __destruct(){
    echo "end";
}

$obj1=new class1();
$obj1->fun1();
?>

```

 PHP OOP Introduction Tutorial in Hindi / Urdu

## Class & Object in PHP

class calculation{  
 public \$a , \$b , \$c ;  
 function sum(){  
 \$this->c = \$this->a + \$this->b;  
 return \$this->c;  
 }  
 function sub(){  
 \$this->c = \$this->a - \$this->b;  
 return \$this->c;  
 }  
}

\$c1 = new calculation();  
\$c1->a = 10;  
\$c1->b = 20;  
echo \$c1->sum() ;  
↓  
30

Yahoo Baba

13:11 / 21:53

www.yahobaba.net

```
constructor.php
```

```
1 <?php
2
3 class person{
4     public $name = "No Name";
5     public $age = 0;
6
7     function __construct($name , $age){
8         $this->name = $name;
9         $this->age = $age;
10    }
11
12    function show(){
13        echo $this->name . " - " . $this->age;
14    }
15 }
16
17 $p1 = new person("Yahoo Baba",20);
18
19 // $p1->name = "Yahoo Baba";
20 // $p1->age = 20;
21
22 $p1->show();
23
```

Yahoo  
Baba

```
employee.php
```

```
3 class employee{
4     public $name;
5     public $age;
6     public $salary;
7
8     function __construct($n, $a, $s){
9         $this->name = $n;
10        $this->age = $a;
11        $this->salary = $s;
12    }
13
14    function info(){
15        echo "<h3>Employee Profile</h3>";
16        echo "Employee Name : " . $this->name . "<br>";
17        echo "Employee Age : " . $this->age . "<br>";
18        echo "Employee Salary : " . $this->salary . "<br>";
19    }
20
21 }
22
23 $e1 = new employee("Ram", 25, 2000);
24
25 $e1->info();      @
26 ?>
```

Yahoo  
Baba



## Overriding Methods

```
class A{
    public $name;
    public function show(){
        echo "My Name" . $this->name;
    }
}

class B extends class A{
    public function show(){
        echo "Your Name" . $this->name;
    }
}
```



## Abstract Class

```
1 → abstract class A{
    protected $name;
    protected function __construct($n){
        $this->name = $n;
    }
}
2 → abstract protected function show(); ← Declare ✓
class B extends class A{
    3 → public function show(){
        echo $this->name;
    }
}
```

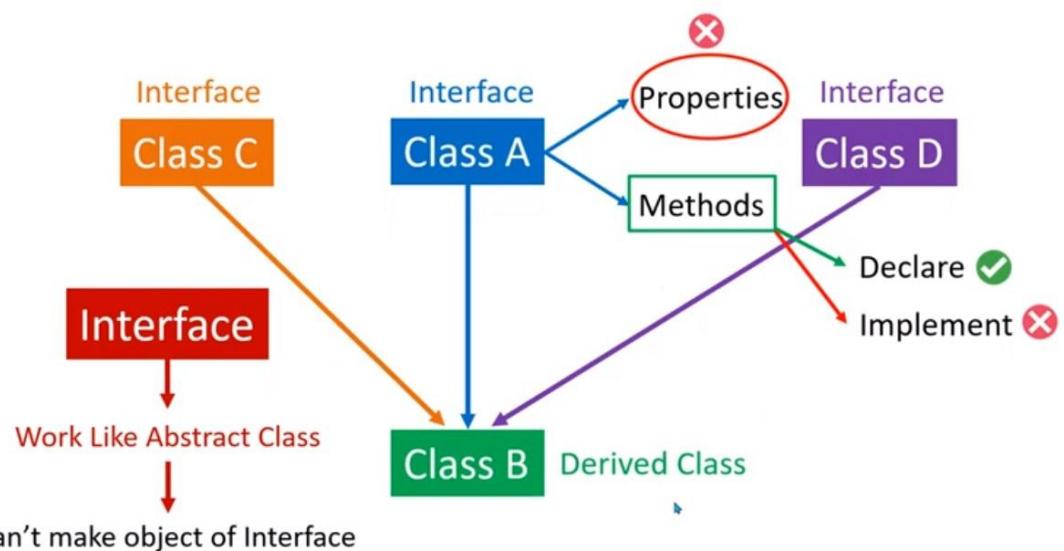
Implement ✗

```
abstract.php
1 <?php
2
3 abstract class parentClass{
4
5     public $name;
6
7     abstract protected function calc($a, $b);
8
9 }
10
11 class childClass extends parentClass{
12
13     public function calc($c, $d){
14         echo $c + $d;
15     }
16
17 }
18
19 $test = new childClass();
20
21 $test->calc(10, 20);
```

Yahoo  
Baba



## What is Interface ?





## Interface

1 → interface A{  
    function hello(\$n);  
}  
  
interface C{  
    2 → function hi(\$n);  
    function bye();  
}

3 → \$test = new A(); 

4 → class B implements A , C{  
    public function hello(\$n){  
        echo "Hello " . \$n;  
    }  
  
    public function hi(\$n){  
        echo "Hi " . \$n;  
    }  
  
    public function bye(){  
        echo "Bye";  
    }  
}



## Static Members in PHP

class personal{ ← Static Class  
    public static \$name = "Yahoo Baba" ;  
    public static function show(){  
        echo self::\$name;  
    }  
}  
  
personal::\$name;  
personal::show();





## Static Members in PHP

```
class personal{  
    public static $name = "Yahoo Baba" ;  
}
```

```
class personal extends accounts{  
    public function show(){  
        echo parent::$name;  
    }  
}
```

```
$test = new accounts();  
$test->show();
```



## Late Static Binding in PHP

```
class personal{  
    protected static $name = "Yahoo" ;  
    public function show(){  
        echo static::name;  
    }  
}  
  
class personal extends accounts{  
    protected static $name = "Baba" ;  
}
```

```
$test = new accounts();  
$test->show();
```

Baba





## Traits in PHP

```
trait test{
    public function hello(){
        echo "Say Hello";
    }
}

class A{
    use test;
}

class B{
    use test;
}
```

```
$obj = new A();
$obj->hello();
```



## What is Type Hinting ?

```
function sum(int $value){
    echo $value + 10;
}

sum(10);
sum("Hey");
```

Type declaration

Valid DataTypes

- Int
- Float
- String
- Bool
- Array
- Class / Interface Name
- Object





## Namespace

First.php

```
namespace first;  
class test{  
}  
}
```

Second.php

```
namespace second;  
class test{  
}
```

Third.php

```
require First.php;  
require second.php;
```

**\$obj = new first\test();**



Yahoo Baba

www.yahooobaba.net

YahOO



## Method Chaining

```
class personal{  
    public function first(){  
        echo "This is first function";  
    }  
  
    public function second(){  
        echo "This is second function";  
    }  
  
    public function third(){  
        echo "This is third function";  
    }  
}
```

```
$test = new personal();  
$test->first();  
$test->second();  
$test->third();
```

**\$test->first()->second()->third();**



Yahoo Baba

www.yahooobaba.net

YahOO



## Magic Methods in PHP

- `__construct()`
- `__destruct()`
- `__get()`
- `__set()`
- `__isset()`
- `__unset()`
- `__autoload()`
- `__clone()`
- `__sleep()`
- `__wakeup()`
- `__call()`
- `__callStatic()`
- `__toString()`
- `__invoke`



## PHP : List of Magic Constants

- `__LINE__`
- `__FILE__`
- `__DIR__`
- `__FUNCTION__`
- `__CLASS__`
- `__METHOD__`
- `__NAMESPACE__`
- `__TRAIT__`



## PHP : List of Conditional Functions

- class\_exists()
- interface\_exists()
- method\_exists()
- trait\_exists()
- property\_exists()
- is\_a()
- is\_subclass\_of()



## PHP : List of Get Functions

- |                        |                           |
|------------------------|---------------------------|
| • get_class            | • get_declared_interfaces |
| • get_parent_class     | • get_declared_traits     |
| • get_class_methods    | • class_alias             |
| • get_class_vars       |                           |
| • get_object_vars      |                           |
| • get_called_class     |                           |
| • get_declared_classes |                           |



## Magic Method

```
<?php
class UserProfile {
    private $data = [];

    public function __construct($name, $email) {
        $this->data['name'] = $name;
        $this->data['email'] = $email;
    }

    public function __get($property) {
        if (array_key_exists($property, $this->data)) {
            return $this->data[$property];
        }
        return null;
    }

    public function __set($property, $value) {
        $this->data[$property] = $value;
    }

    public function __toString() {
        return "Name: {$this->data['name']}, Email: {$this->data['email']}";
    }
}

// Create an instance of UserProfile
$user = new UserProfile("John Doe", "johndoe@example.com");

// Access and modify properties using magic methods
echo "Name: " . $user->name . "\n"; // __get is called
$user->name = "Jane Doe"; // __set is called
echo "Updated Name: " . $user->name . "\n";

// Convert the object to a string using __toString
echo "User Profile: " . $user . "\n";
?>
```

1. We have a UserProfile class that stores user data in a private \$data array.

2. The \_\_construct method is called when an object of the class is created, initializing the name and email properties.

3.The \_\_get method is used to retrieve property values. If a property doesn't exist, it returns null.

4.The \_\_set method allows us to set or modify property values.

5.The \_\_toString method customizes how the object is converted to a string when we use it in a string context, like echo.

```
<?php
class MagicDemo {
    private $data = [];

    public function __construct() {
        echo "An instance of MagicDemo has been created.\n";
    }

    public function __destruct() {
        echo "An instance of MagicDemo is being destroyed.\n";
    }

    public function __get($property) {
        echo "Getting property '$property': ";
        if (array_key_exists($property, $this->data)) {
            return $this->data[$property];
        } else {
            return "Property does not exist.\n";
        }
    }

    public function __set($property, $value) {
        echo "Setting property '$property' to '$value'.\n";
        $this->data[$property] = $value;
    }

    public function __isset($property) {
        echo "Checking if property '$property' is set: ";
        return isset($this->data[$property]);
    }

    public function __unset($property) {
        echo "Unsetting property '$property'.\n";
        unset($this->data[$property]);
    }

    public function __call($method, $arguments) {
        echo "Calling method '$method' with arguments: ";
        print_r($arguments);
    }
}
```

```
}

public function __toString() {
    return "This is a MagicDemo object.";
}
}

// Create an instance of MagicDemo
$obj = new MagicDemo();

// Access properties using magic __get and __set
$obj->name = "John";
echo "Name: " . $obj->name . "\n";

// Check property existence using __isset
echo isset($obj->age) ? "Age is set.\n" : "Age is not set.\n";

// Unset property using __unset
unset($obj->name);

// Call undefined method using __call
$obj->doSomething(42, "Hello");

// Convert object to string using __toString
echo $obj;
?>
```

## ALL php magic constant uses one program

1. `__LINE__`: Returns the current line number of the file.
2. `__FILE__`: Returns the full path and filename of the file.
3. `__DIR__`: Returns the directory of the file.
4. `__FUNCTION__`: Returns the name of the current function.
5. `__CLASS__`: Returns the name of the current class.
6. `__METHOD__`: Returns the name of the current class method.
7. `__NAMESPACE__`: Returns the current namespace name

```
<?php
class MyClass {
    public function __construct() {
        echo "Inside " . __METHOD__ . " in " . __CLASS__ . " within namespace
" . __NAMESPACE__ . "\n";
    }

    public function showFilePath() {
        echo "Current file: " . __FILE__ . "\n";
        echo "Current directory: " . __DIR__ . "\n";
    }
}

echo "Current line number: " . __LINE__ . "\n";

$obj = new MyClass();
$obj->showFilePath();
?>
```

1. `__LINE__` is used to display the current line number.
2. `__FILE__` and `__DIR__` are used within the `showFilePath` method of the `MyClass` class to display the current file and directory.
3. `__METHOD__`, `__CLASS__`, and `__NAMESPACE__` are used within the constructor of the `MyClass` class to display information about the method, class, and namespace.

```

<?php
namespace MyNamespace;

class MyClass {
    private $property;

    public function __construct($value) {
        $this->property = $value;
    }

    public function displayInfo() {
        echo "Inside " . __METHOD__ . " in " . __CLASS__ . " within namespace
" . __NAMESPACE__ . "\n";
    }

    public function showFilePath() {
        echo "Current file: " . __FILE__ . "\n";
        echo "Current directory: " . __DIR__ . "\n";
    }

    public function getProperty() {
        return $this->property;
    }
}

// Use __LINE__ to display the current line number
echo "Current line number: " . __LINE__ . "\n";

// Create an instance of MyClass and use various magic constants
$obj = new MyClass(42);
$obj->displayInfo();
$obj->showFilePath();

// Access class property and display it along with magic constants
echo "Property value: " . $obj->getProperty() . "\n";
?>

```

1. `__LINE__` is used to display the current line number.
2. `__METHOD__`, `__CLASS__`, and `__NAMESPACE__` are used within the `MyClass` class to display information about the method, class, and namespace.
3. `__FILE__` and `__DIR__` are used within the `showFilePath` method of the `MyClass` class to display the current file and directory.
4. Magic constants are used to display information within the main part of the script.

## let var const in js program example

```
// Using var
function varExample() {
    var x = 10;
    if (true) {
        var x = 20; // This reassigns the same variable x
    }
    console.log("Var Example:", x); // Outputs 20
}
varExample();

// Using let
function letExample() {
    let y = 10;
    if (true) {
        let y = 20; // This creates a new variable y within the block
    }
    console.log("Let Example:", y); // Outputs 10
}
letExample();

// Using const
function constExample() {
    const z = 10;
    // z = 20; // This will result in an error (assignment to a constant
    // variable)
    console.log("Const Example:", z);
}
constExample();
```

1.The varExample function uses var to declare a variable x. It reassigns x within an if block, and the modified value is reflected outside the block.

2.The letExample function uses let to declare a variable y. It creates a new variable y within the if block, and the original value remains unchanged outside the block.

3.The constExample function uses const to declare a constant variable z. An attempt to reassign z to a different value would result in an error.

## Calling parent constructors

```
class ParentClass {
    public function __construct() {
        echo "ParentClass constructor called<br>";
    }
}

class ChildClass extends ParentClass {
    public function __construct() {
        parent::__construct(); // Calling the parent class constructor
        echo "ChildClass constructor called<br>";
    }
}

$child = new ChildClass();
```

```
class Name {
    var $_firstName;
    var $_lastName;

    function Name($first_name, $last_name) {
        $this->_firstName = $first_name;
        $this->_lastName = $last_name;
    }

    function toString() {
        return($this->_lastName . ", " . $this->_firstName);
    }
}
class NameSub1 extends Name {
    var $_middleInitial;

    function NameSub1($first_name, $middle_initial, $last_name) {
        Name::Name($first_name, $last_name);
        $this->_middleInitial = $middle_initial;
    }

    function toString() {
        return(Name::toString() . " " . $this->_middleInitial);
    }
}
```

## Calling Parent Constructor

```
class DatabaseConnection {
    protected $host;
    protected $username;
    protected $password;
    protected $database;
    protected $connection;

    public function __construct($host, $username, $password, $database) {
        $this->host = $host;
        $this->username = $username;
        $this->password = $password;
        $this->database = $database;
    }

    public function connect() {
        // Generic database connection logic
        $this->connection = new mysqli($this->host, $this->username, $this-
>password, $this->database);
        if ($this->connection->connect_error) {
            die("Database Connection Error: " . $this->connection-
>connect_error);
        }
        echo "Connected to the database<br>";
    }
}

class MySQLDatabaseConnection extends DatabaseConnection {
    public function __construct($host, $username, $password, $database) {
        parent::__construct($host, $username, $password, $database);
    }

    // Additional methods and MySQL-specific logic can be added here
}

// Example usage
$mysqlDb = new MySQLDatabaseConnection('localhost', 'mysqluser', 'mysqlpass',
'mysqldb');
$mysqlDb->connect();
```

## static Keyword

```
<?php
class Foo {
    public static $my_static = 'foo';

    public function staticValue() {
        return self::$my_static;
    }
}

print Foo::$my_static . "\n";
$foo = new Foo();

print $foo->staticValue() . "\n";
?>
```

## late static binding

```
class ParentClass {
    public static function whoAmI() {
        echo "I am from " . static::class . "<br>";
    }
}

class ChildClass extends ParentClass {}

class AnotherChildClass extends ParentClass {}

ParentClass::whoAmI();      // Output: I am from ParentClass
ChildClass::whoAmI();       // Output: I am from ChildClass
AnotherChildClass::whoAmI(); // Output: I am from AnotherChildClass
```

## Realworld example

```
class Shape {
    protected $area;

    public function getArea() {
        return $this->area;
    }

    public static function create($type, $params) {
        switch ($type) {
            case 'circle':
                return Circle::createCircle($params);
            case 'rectangle':
                return Rectangle::createRectangle($params);
            default:
                throw new InvalidArgumentException("Invalid shape type: $type");
        }
    }
}

class Circle extends Shape {
    protected $radius;

    public function __construct($radius) {
        $this->radius = $radius;
        $this->area = $this->calculateArea();
    }

    public function calculateArea() {
        return pi() * $this->radius * $this->radius;
    }

    public static function createCircle($params) {
        return new static($params['radius']);
    }
}

class Rectangle extends Shape {
    protected $width;
    protected $height;

    public function __construct($width, $height) {
        $this->width = $width;
        $this->height = $height;
    }
}
```

```

        $this->area = $this->calculateArea();
    }

    public function calculateArea() {
        return $this->width * $this->height;
    }

    public static function createRectangle($params) {
        return new static($params['width'], $params['height']);
    }
}

// Create a circle and a rectangle using the factory method
$circle = Shape::create('circle', ['radius' => 5]);
$rectangle = Shape::create('rectangle', ['width' => 4, 'height' => 6]);

// Display the areas of the shapes
echo "Circle Area: " . $circle->getArea() . "<br>";
echo "Rectangle Area: " . $rectangle->getArea() . "<br>";

```

## parent Keyword

```

class ParentClass {
    protected $property = "I am a property in the parent class";

    public function printProperty() {
        echo $this->property . "<br>";
    }
}

class ChildClass extends ParentClass {
    protected $property = "I am a property in the child class";

    public function printParentProperty() {
        // Access the parent class property using the parent keyword
        echo parent::$property . "<br>";

        // Call the parent class method using the parent keyword
        parent::printProperty();
    }
}

$child = new ChildClass();
$child->printParentProperty();

```

## **final**

- **final** keyword is used to create final method or final class.
- A final method cannot be overridden in child class
- A final class cannot be inherited It means we can not create sub class of a final class.

```
Class Father {  
    function display(){  
        echo "You can override me becoz I am not final";  
    }  
    final function show(){  
        echo "I am final so you cannot override me";  
    }  
}  
class Son extends Father {  
    function display(){  
        echo "Yea I overrided";  
    }  
}
```

## **Final Class**

```
final class Father {  
    function display(){  
        echo "Final";  
    }  
}
```

## Final Keword (method)

```
class ParentClass {
    final protected function getProperty() {
        return "This is a final property.";
    }
}

class ChildClass extends ParentClass {
    public function getProperty() {
        return parent::getProperty();
    }
}

$child = new ChildClass();
echo $child->getProperty(); // Output: This is a final property.
```

## Final Keword (class)

```
final class FinalClass {
    public function sayHello() {
        echo "Hello from FinalClass!<br>";
    }
}

// Attempting to extend a final class will result in an error
class ChildClass extends FinalClass {
    // This will cause a fatal error
}

$finalObj = new FinalClass();
$finalObj->sayHello();
```

## Abstract class and method

```
// Abstract class for PaymentGateway
abstract class PaymentGateway {
    protected $apiKey;

    public function __construct($apiKey) {
        $this->apiKey = $apiKey;
    }

    // Abstract method for processing payment
    abstract public function processPayment($amount);
}

// Concrete class for PayPal payment gateway
class PayPalGateway extends PaymentGateway {
    public function processPayment($amount) {
        // Implement the PayPal-specific payment processing logic here
        // Use $this->apiKey to authenticate with PayPal API
        // Process the payment and return the result
        echo "Processing payment of $amount USD using PayPal.<br>";
    }
}

// Concrete class for Stripe payment gateway
class StripeGateway extends PaymentGateway {
    public function processPayment($amount) {
        // Implement the Stripe-specific payment processing logic here
        // Use $this->apiKey to authenticate with Stripe API
        // Process the payment and return the result
        echo "Processing payment of $amount USD using Stripe.<br>";
    }
}

// Usage
$paypalGateway = new PayPalGateway('your_paypal_api_key');
$stripeGateway = new StripeGateway('your_stripe_api_key');

$paypalGateway->processPayment(100);
$stripeGateway->processPayment(75);
```

## Other example

```
abstract class PaymentGateway {
    abstract public function processPayment($amount);
}

class PayPalGateway extends PaymentGateway {
    public function processPayment($amount) {
        // Process payment using PayPal API
        echo "Processing payment of $amount via PayPal.<br>";
    }
}

class StripeGateway extends PaymentGateway {
    public function processPayment($amount) {
        // Process payment using Stripe API
        echo "Processing payment of $amount via Stripe.<br>";
    }
}

// Example usage
$paypalGateway = new PayPalGateway();
$paypalGateway->processPayment(100);

$stripeGateway = new StripeGateway();
$stripeGateway->processPayment(150);
```

## Interface Example

```
1 Interface
2
3 => Interface resamble abstract classes
4 => Interface contains abstract function like abstract classes
5 => Interface can't be instantiated like abstract classes
6 => Interface can not contain concrete methods like abstract classes
7 => Interface contains only public methods not private or protected
8 => An abstract can only extends by 1 abstract class
9 => An class can implements more than 1 interface
10 => Multiple inheritance can be achived through interfaces
11 => Interface can not contains constructor
12 => Interface can not contains data members
13
14
```

---

```
<?php

interface demo
{
    I
    public function test1();
    public function test2();
}

interface demo2
{
    public function test3();
}

class childClass implements demo,demo2
{
    public function test1(){}
    public function test2(){}
}
```

```
interface DatabaseInterface {
    public function connect($host, $username, $password, $database);
    public function query($sql);
    public function close();
}

class MySQLDatabase implements DatabaseInterface {
    private $connection;

    public function connect($host, $username, $password, $database) {
        // Implement the connection logic for MySQL
        $this->connection = mysqli_connect($host, $username, $password,
$database);
        if (!$this->connection) {
            die("MySQL Connection Error: " . mysqli_connect_error());
        }
    }

    public function query($sql) {
        // Implement the query execution logic for MySQL
        $result = mysqli_query($this->connection, $sql);
        return $result;
    }

    public function close() {
        // Implement the connection closing logic for MySQL
        mysqli_close($this->connection);
    }
}

class PostgreSQLDatabase implements DatabaseInterface {
    private $connection;

    public function connect($host, $username, $password, $database) {
        // Implement the connection logic for PostgreSQL
        $this->connection = pg_connect("host=$host dbname=$database
user=$username password=$password");
        if (!$this->connection) {
            die("PostgreSQL Connection Error: " . pg_last_error());
        }
    }

    public function query($sql) {
        // Implement the query execution logic for PostgreSQL
        $result = pg_query($this->connection, $sql);
    }
}
```

```

        return $result;
    }

    public function close() {
        // Implement the connection closing logic for PostgreSQL
        pg_close($this->connection);
    }
}

// Example usage
$mysqlDB = new MySQLDatabase();
$mysqlDB->connect('localhost', 'user', 'password', 'mydb');
$result = $mysqlDB->query('SELECT * FROM mytable');
$mysqlDB->close();

$postgresDB = new PostgreSQLDatabase();
$postgresDB->connect('localhost', 'user', 'password', 'mydb');
$result = $postgresDB->query('SELECT * FROM mytable');
$postgresDB->close();

```

## Another Example Interface

```

interface ContentElement {
    public function render();
    public function getMetadata();
}

class Article implements ContentElement {
    private $title;
    private $content;

    public function __construct($title, $content) {
        $this->title = $title;
        $this->content = $content;
    }

    public function render() {
        // Implement rendering logic for an article
        return "<h1>{$this->title}</h1><p>{$this->content}</p>";
    }

    public function getMetadata() {

```

```
// Implement metadata retrieval logic for an article
    return "Title: {$this->title}";
}
}

class Image implements ContentElement {
    private $src;
    private $alt;

    public function __construct($src, $alt) {
        $this->src = $src;
        $this->alt = $alt;
    }

    public function render() {
        // Implement rendering logic for an image
        return "<img src='{$this->src}' alt='{$this->alt}'>";
    }

    public function getMetadata() {
        // Implement metadata retrieval logic for an image
        return "Alt Text: {$this->alt}";
    }
}

// Example usage
$article = new Article("Sample Article", "This is a sample article
content.");
$image = new Image("image.jpg", "Sample Image Alt Text");

echo $article->render();
echo $article->getMetadata();

echo $image->render();
echo $image->getMetadata();
```

## PHP interface example code in details in

In PHP, an interface is a contract that defines a set of methods a class must implement. Interfaces allow you to define a common set of methods that multiple classes can adhere to. Here's an example of a PHP interface along with a class that implements it:

```
<?php

// Define an interface
interface Shape {
    public function getArea();
    public function getPerimeter();
}

// Create a class that implements the interface
class Circle implements Shape {
    private $radius;

    public function __construct($radius) {
        $this->radius = $radius;
    }

    public function getArea() {
        return pi() * pow($this->radius, 2);
    }

    public function getPerimeter() {
        return 2 * pi() * $this->radius;
    }
}

// Create another class that implements the interface
class Rectangle implements Shape {
    private $width;
    private $height;

    public function __construct($width, $height) {
        $this->width = $width;
        $this->height = $height;
    }

    public function getArea() {
        return $this->width * $this->height;
    }

    public function getPerimeter() {
```

```

        return 2 * ($this->width + $this->height);
    }
}

// Create instances of the classes and use the interface methods
$circle = new Circle(5);
echo "Circle Area: " . $circle->getArea() . "\n";
echo "Circle Perimeter: " . $circle->getPerimeter() . "\n";

$rectangle = new Rectangle(4, 6);
echo "Rectangle Area: " . $rectangle->getArea() . "\n";
echo "Rectangle Perimeter: " . $rectangle->getPerimeter() . "\n";
?>

```

## Overloading in php

```

class Calculator {
    public function calculate($a, $b = null, $c = null) {
        if ($c !== null) {
            // Three arguments provided, perform a three-operand calculation
            return $a + $b + $c;
        } elseif ($b !== null) {
            // Two arguments provided, perform a two-operand calculation
            return $a + $b;
        } else {
            // Only one argument provided, return the argument itself
            return $a;
        }
    }
}

$calculator = new Calculator();

// Example usages of method "calculate"
$result1 = $calculator->calculate(5);           // Returns 5
$result2 = $calculator->calculate(5, 3);         // Returns 8
$result3 = $calculator->calculate(5, 3, 2);       // Returns 10

echo "Result 1: $result1<br>";
echo "Result 2: $result2<br>";
echo "Result 3: $result3<br>";

```

## Another example

```
class Database {
    private $connection;

    public function __construct($host, $username, $password, $database) {
        // Establish a database connection (you would use your preferred
        database extension)
        $this->connection = new mysqli($host, $username, $password,
$database);
        if ($this->connection->connect_error) {
            die("Database Connection Error: " . $this->connection-
>connect_error);
        }
    }

    public function query($sql, $operation = 'SELECT', $params = []) {
        switch ($operation) {
            case 'SELECT':
                return $this->selectQuery($sql, $params);
            case 'INSERT':
                return $this->insertQuery($sql, $params);
            case 'UPDATE':
                return $this->updateQuery($sql, $params);
            default:
                throw new InvalidArgumentException("Unsupported database
operation: $operation");
        }
    }

    private function selectQuery($sql, $params) {
        // Implement SELECT query logic with prepared statement
        // Execute the query, bind parameters, fetch results, etc.
        return "SELECT result for query: $sql";
    }

    private function insertQuery($sql, $params) {
        // Implement INSERT query logic with prepared statement
        // Execute the query, bind parameters, handle insert IDs, etc.
        return "INSERT result for query: $sql";
    }

    private function updateQuery($sql, $params) {
        // Implement UPDATE query logic with prepared statement
```

```
// Execute the query, bind parameters, handle affected rows, etc.
    return "UPDATE result for query: $sql";
}

public function close() {
    // Close the database connection
    $this->connection->close();
}
}

// Example usage
$db = new Database('localhost', 'username', 'password', 'mydb');

$selectResult = $db->query('SELECT * FROM users WHERE id = ?', 'SELECT',
[1]);
$insertResult = $db->query('INSERT INTO orders (product, quantity) VALUES (?, ?)', 'INSERT', ['Widget', 5]);
$updateResult = $db->query('UPDATE products SET price = ? WHERE id = ?', 'UPDATE', [19.99, 123]);

$db->close();

echo $selectResult . "<br>";
echo $insertResult . "<br>";
echo $updateResult . "<br>";
```

## Another example

```
class Calculator {
    public function calculate($operation, ...$operands) {
        switch ($operation) {
            case 'add':
                return $this->add(...$operands);
            case 'subtract':
                return $this->subtract(...$operands);
            case 'multiply':
                return $this->multiply(...$operands);
            default:
                throw new InvalidArgumentException("Unsupported operation: $operation");
        }
    }

    private function add($a, $b) {
        return $a + $b;
    }

    private function subtract($a, $b) {
        return $a - $b;
    }

    private function multiply($a, $b) {
        return $a * $b;
    }
}

$calculator = new Calculator();

// Example usages
$result1 = $calculator->calculate('add', 5, 3);
$result2 = $calculator->calculate('subtract', 10, 2);
$result3 = $calculator->calculate('multiply', 4, 6);

echo "Addition result: $result1<br>";
echo "Subtraction result: $result2<br>";
echo "Multiplication result: $result3<br>";
```

## Overriding example

```
class Content {
    protected $title;
    protected $content;

    public function __construct($title, $content) {
        $this->title = $title;
        $this->content = $content;
    }

    public function display() {
        // Generic content display
        echo "<h2>{$this->title}</h2>";
        echo "<p>{$this->content}</p>";
    }
}

class Article extends Content {
    private $author;

    public function __construct($title, $content, $author) {
        parent::__construct($title, $content);
        $this->author = $author;
    }

    public function display() {
        // Display article-specific content
        echo "<h2>{$this->title}</h2>";
        echo "<p>{$this->content}</p>";
        echo "<p>Author: {$this->author}</p>";
    }
}

class Image extends Content {
    private $imagePath;

    public function __construct($title, $content, $imagePath) {
        parent::__construct($title, $content);
        $this->imagePath = $imagePath;
    }

    public function display() {
        // Display image-specific content
    }
}
```

```

        echo "<h2>{$this->title}</h2>";
        echo "<img src='{$this->imagePath}' alt='{$this->title}'><br>";
        echo "<p>{$this->content}</p>";
    }
}

// Example usage
$article = new Article("Sample Article", "This is a sample article content.", "John Doe");
$image = new Image("Sample Image", "This is a sample image description.", "image.jpg");

$article->display();
$image->display();

```

## Another Example

```

class DatabaseConnection {
    protected $host;
    protected $username;
    protected $password;
    protected $database;

    public function __construct($host, $username, $password, $database) {
        $this->host = $host;
        $this->username = $username;
        $this->password = $password;
        $this->database = $database;
    }

    public function connect() {
        // Generic database connection logic
        $this->connection = new mysqli($this->host, $this->username, $this->password, $this->database);
        if ($this->connection->connect_error) {
            die("Database Connection Error: " . $this->connection->connect_error);
        }
        echo "Connected to the database using generic driver<br>";
    }
}

```

```

class MySQLDatabaseConnection extends DatabaseConnection {
    public function connect() {
        // MySQL-specific database connection logic
        parent::connect(); // Call the parent class's connect method first
        echo "Connected to the MySQL database using MySQLi driver<br>";
    }
}

// Example usage
$mysqlDb = new MySQLDatabaseConnection('localhost', 'mysqluser', 'mysqlpass',
'mysqldb');
$mysqlDb->connect();

```

## Overriding very sample example

```

class Animal {
    public function speak() {
        echo "Animal speaks!";
    }
}

class Dog extends Animal {
    public function speak() {
        echo "Dog barks!";
    }
}

$animal = new Animal();
$dog = new Dog();

$animal->speak(); // Output: Animal speaks!
$dog->speak();   // Output: Dog barks!

```

## Error handling in php

```
// Define a custom error handler function
function customErrorHandler($errno, $errstr, $errfile, $errline) {
    echo "Custom Error Handler: [$errno] $errstr\n";
    echo "Error on line $errline in $errfile\n";
}

// Set the custom error handler
set_error_handler("customErrorHandler");

try {
    // Simulate a division by zero error
    $result = 10 / 0;
} catch (Exception $e) {
    // This block will not be executed for PHP errors, only for exceptions
    echo "Caught Exception: " . $e->getMessage() . "\n";
} finally {
    // This block will always be executed, whether there was an error or not
    echo "Finally block executed\n";
}

// Attempt to open a non-existent file
$file = fopen("nonexistent.txt", "r");
if ($file === false) {
    echo "Error: Failed to open the file.\n";
}

// Trigger a user-defined error
trigger_error("This is a custom error message.", E_USER_WARNING);

// Attempt to call a non-existent function
nonExistentFunction();

// Restore the default error handler
restore_error_handler();
```

## Error handling example

```
class DatabaseConnection {
    protected $host;
    protected $username;
    protected $password;
    protected $database;
    protected $connection;

    public function __construct($host, $username, $password, $database) {
        $this->host = $host;
        $this->username = $username;
        $this->password = $password;
        $this->database = $database;

        // Attempt to establish a database connection
        $this->connect();
    }

    private function connect() {
        // Establish the database connection
        $this->connection = new mysqli($this->host, $this->username, $this-
>password, $this->database);

        // Check for connection errors
        if ($this->connection->connect_error) {
            throw new Exception("Database Connection Error: " . $this-
>connection->connect_error);
        }
    }

    public function query($sql) {
        // Perform a database query
        $result = $this->connection->query($sql);

        // Check for query errors
        if ($result === false) {
            throw new Exception("Database Query Error: " . $this->connection-
>error);
        }

        return $result;
    }
}
```

```

    public function close() {
        // Close the database connection
        $this->connection->close();
    }
}

// Example usage with error handling
try {
    $db = new DatabaseConnection('localhost', 'username', 'password',
'mydb');

    // Example query with an error (table doesn't exist)
    $result = $db->query('SELECT * FROM non_existing_table');

    // Close the database connection
    $db->close();
} catch (Exception $e) {
    // Handle database-related exceptions
    echo "Caught exception: " . $e->getMessage();
}

```

## Another Example

```

class DatabaseConnection {
    protected $host;
    protected $username;
    protected $password;
    protected $database;
    protected $connection;

    public function __construct($host, $username, $password, $database) {
        $this->host = $host;
        $this->username = $username;
        $this->password = $password;
        $this->database = $database;

        // Attempt to establish a database connection
        $this->connect();
    }

    private function connect() {
        // Establish the database connection
        $this->connection = new mysqli($this->host, $this->username, $this-
>password, $this->database);
    }
}

```

```

    // Check for connection errors
    if ($this->connection->connect_error) {
        throw new Exception("Database Connection Error: " . $this-
>connection->connect_error);
    }
}

public function executeQuery($sql) {
    // Execute a database query and return the result
    return $this->connection->query($sql);
}

public function getLastInsertId() {
    // Get the last auto-generated ID from an INSERT query
    return $this->connection->insert_id;
}

public function close() {
    // Close the database connection
    $this->connection->close();
}
}

class User {
    private $db;

    public function __construct(DatabaseConnection $db) {
        $this->db = $db;
    }

    public function addUser($username, $email, $password) {
        $username = $this->db->connection->real_escape_string($username);
        $email = $this->db->connection->real_escape_string($email);
        $password = $this->db->connection->real_escape_string($password);

        $sql = "INSERT INTO users (username, email, password) VALUES
('{$username}', '{$email}', '{$password}')";

        $result = $this->db->executeQuery($sql);

        if ($result) {
            echo "User added successfully with ID: " . $this->db-
>getLastInsertId();
        } else {
    }
}

```

```

        throw new Exception("Error adding user: " . $this->db-
>connection->error);
    }
}
}

// Example usage
try {
    $db = new DatabaseConnection('localhost', 'username', 'password',
'mydb');
    $user = new User($db);

    $user->addUser('john_doe', 'john@example.com', 'password123');

    // Close the database connection
    $db->close();
} catch (Exception $e) {
    // Handle exceptions, e.g., display an error message
    echo "Caught exception: " . $e->getMessage();
}

```

## Exception handling in php

```

class DatabaseConnectionException extends Exception {
    public function errorMessage() {
        return "Database Connection Error: " . $this->getMessage();
    }
}

class Database {
    private $connection;

    public function __construct($host, $username, $password, $database) {
        try {
            $this->connection = new mysqli($host, $username, $password,
$database);

            if ($this->connection->connect_error) {
                throw new DatabaseConnectionException($this->connection-
>connect_error);
            }
        } catch (DatabaseConnectionException $e) {
            // Handle the exception
        }
    }
}

```

```

        die($e->errorMessage());
    }
}

public function query($sql) {
    try {
        if (!$this->connection) {
            throw new DatabaseConnectionException("No database
connection.");
        }

        $result = $this->connection->query($sql);

        if ($result === false) {
            throw new Exception("Database query error: " . $this-
>connection->error);
        }

        return $result;
    } catch (Exception $e) {
        // Handle the exception
        die("Error: " . $e->getMessage());
    }
}

public function close() {
    if ($this->connection) {
        $this->connection->close();
    }
}

// Example usage
try {
    $db = new Database("localhost", "username", "password", "mydb");
    $result = $db->query("SELECT * FROM non_existing_table");
    $db->close();
} catch (Exception $e) {
    // Handle any exception thrown during database connection or query
    echo "Caught exception: " . $e->getMessage();
}

```

# Google login API

Google Sign-In with PHP using Google API Client Library:

## 1. Create a Project in Google Cloud Console:

- Go to the [Google Cloud Console](#).
- Create a new project or select an existing one.
- Enable the "Google+ API" for your project.

## 2. Create OAuth 2.0 Credentials:

- In the Cloud Console, navigate to "APIs & Services" > "Credentials."
- Click on "Create Credentials" and choose "OAuth client ID."
- Select "Web application" as the application type.
- Configure the authorized JavaScript origins and redirect URIs. For local development, you can use "<http://localhost>" as the origin.
- Click "Create" to generate your OAuth client ID and client secret.

## 3. Install Google API Client Library for PHP:

- You can install the Google API Client Library for PHP using Composer:

```
composer require google/apiclient:^2.0
```

## 4. Set Up the PHP Code:

- Create a PHP file (e.g., `google_login.php`) and add the following code:

```
<?php
require_once 'vendor/autoload.php';

$client = new Google_Client();
$client->setClientId('YOUR_CLIENT_ID');
$client->setClientSecret('YOUR_CLIENT_SECRET');
$client->setRedirectUri('YOUR_REDIRECT_URI');
$client->addScope('email');
$client->addScope('profile');

if (isset($_GET['code'])) {
    $token = $client->fetchAccessTokenWithAuthCode($_GET['code']);
    $client->setAccessToken($token);

    $oauth2Service = new Google_Service_Oauth2($client);
    $userInfo = $oauth2Service->userinfo->get();

    // Now, you can use $userInfo to access user data (e.g., $userInfo->getEmail(), $userInfo->getGivenName()).
}
```

```
echo "Welcome, " . $userInfo->getName();
} else {
    $authUrl = $client->createAuthUrl();
    echo "<a href='$authUrl'>Login with Google</a>";
}
?>
```

**For insert data in MySQL first we have to create a table in data base.**

Here we using 3 file for insert data in MySQL:

```
CREATE TABLE `employee` (
    `userid` int(8) NOT NULL AUTO_INCREMENT,
    `first_name` varchar(55) NOT NULL,
    `last_name` varchar(55) NOT NULL,
    `city_name` varchar(55) NOT NULL,
    `email` varchar(50) NOT NULL,
    `datetime` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

**index.php:**for getting the values from the user

```
<!DOCTYPE html>
<html>
    <body>
        <form method="post" action="process.php">
            First name:<br>
            <input type="text" name="first_name">
            <br>
            Last name:<br>
            <input type="text" name="last_name">
            <br>
            City name:<br>
            <input type="text" name="city_name">
            <br>
            Email Id:<br>
            <input type="email" name="email">
            <br><br>
            <input type="submit" name="submit" value="submit">
        </form>
    </body>
</html>
```

**Crud.php:**For connecting data base and function

```
<?php
    class Crud
    {
        private $servername = "localhost";
        private $username   = "root";
        private $password   = "";
        private $dbname     = "oops_db";
        public $con;
        public $employeeTable = "employee";
        public function __construct()
        {
            try {
                $this->con = new mysqli($this->servername, $this->username,
                $this->password, $this->dbname);
            } catch (Exception $e) {
                echo $e->getMessage();
            }
        }
        public function save($first_name, $last_name, $city_name, $email,
        $insertdate)
        {
            $sql = "INSERT INTO $this-
>employeeTable(first_name,last_name,city_name,email,datetime)
VALUES('$first_name', '$last_name','$city_name','$email','$insertdate')";
            $query = $this->con->query($sql);
            if ($query) {
                return true;
            }else{
                return false;
            }
        }
    }
?>
```

**process.php:A PHP file that process the request**

```
<?php
    include_once("Crud.php");
    $insertdata=new Crud();
    if(isset($_POST['submit']))
    {
        $first_name = $_POST['first_name'];
        $last_name = $_POST['last_name'];
        $city_name = $_POST['city_name'];
        $email = $_POST['email'];
        date_default_timezone_set("Asia/Calcutta");
        $insertdate = date("Y-m-d H:i:s");
        $sql=$insertdata->save($first_name,
        $last_name,$city_name,$email,$insertdate);
        if($sql)
        {
            echo "Data inserted successfully !";
        }
        else
        {
            echo "Data inserted error !";
        }
    }
?>
```

**For retrieve data from MySQL first we have to create a table in data base.**

```
CREATE TABLE `employee` (
    `userid` int(8) NOT NULL AUTO_INCREMENT,
    `first_name` varchar(55) NOT NULL,
    `last_name` varchar(55) NOT NULL,
    `city_name` varchar(55) NOT NULL,
    `email` varchar(50) NOT NULL,
    `datetime` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
crud.php
<?php
    class Crud
    {
        private $servername = "localhost";
        private $username = "root";
        private $password = "";
        private $dbname = "oops_db";
        public $con;
        public $employeeTable = "employee";
        public function __construct()
        {
            try {
                $this->con = new mysqli($this->servername, $this->username,
                $this->password, $this->dbname);
            } catch (Exception $e) {
                echo $e->getMessage();
            }
        }
        /* Fetch employee records for show listing */
        public function displayRecord()
        {
            $sql = "SELECT * FROM $this->employeeTable";
            $query = $this->con->query($sql);
            $data = array();
            if ($query->num_rows > 0) {
                while ($row = $query->fetch_assoc()) {
                    $data[] = $row;
                }
                return $data;
            }else{
                return false;
            }
        }
    }
?>
```

```
view.php
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Bootstrap Example</title>
    <meta charset="utf-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"></script>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</head>
<body>
<?php
include_once("Crud.php");
$viewdata=new Crud();
$customers = $viewdata->displayRecord();
$output ="";
$output .="<table class='table table-striped table-hover'>
<thead>
<tr>
<th>Id</th>
<th>First Name</th>
<th>Last Name</th>
<th>City Name</th>
<th>Email</th>

</tr>
</thead>
<tbody>";
foreach ($customers as $customer) {
$output.= "<tr>
<td>".$customer['userid']."'</td>
<td>".$customer['first_name']."'</td>
<td>".$customer['last_name']."'</td>
<td>".$customer['city_name']."'</td>
<td>".$customer['email']."'</td>

</tr>";
}
$output .= "</tbody>
</table>";
echo $output;
?>
</body></html>
```

## PHP oops update data

*crud.php*

```
<?php

    class Crud
    {
        private $servername = "localhost";
        private $username   = "root";
        private $password   = "";
        private $dbname     = "oops_db";
        public $con;
        public $employeeTable = "employee";
        public function __construct()
        {
            try {
                $this->con = new mysqli($this->servername, $this->username, $this->password,
                $this->dbname);
            } catch (Exception $e) {
                echo $e->getMessage();
            }
        }

        public function update($userid,$first_name,
        $last_name,$city_name,$email,$insertdate)
        {
            echo $sql = "UPDATE $this->employeeTable SET
            first_name='$first_name',last_name='$last_name',city_name='$city_name',email='$em
            ail' where userid=$userid";
            $query = $this->con->query($sql);
            if ($query) {
                return true;
            }else{
                return false;
            }
        }

    }
?>
```

## *update.php*

```
<?php
    include_once("Crud.php");
    $userid=$_GET['userid'];
    $viewdata=new Crud();
    $employee = $viewdata->displayRecordbyid($userid);
?
<!DOCTYPE html>
<html>
<body>
<form method="post" action="update_process.php">
First name:<br>
<input type="text" name="first_name" value="<?php echo
$employee['first_name'];?>">
<br>
Last name:<br>
<input type="text" name="last_name" value="<?php echo
$employee['last_name'];?>">
<br>
City name:<br>
<input type="text" name="city_name" value="<?php echo
$employee['city_name'];?>">
<br>
Email Id:<br>
<input type="email" name="email" value="<?php echo
$employee['email'];?>">
<br><br>
<input type="hidden" name="userid" value="<?php echo
$employee['userid'];?>">
<input type="submit" name="submit" value="submit">
</form>
</body>
</html>
```

## *update\_process.php*

```
<?php
    include_once("Crud.php");
    $insertdata=new Crud();
    if(isset($_POST['submit']))
    {
        $userid = $_POST['userid'];
        $first_name = $_POST['first_name'];
        $last_name = $_POST['last_name'];
    }
}
```

```

$city_name = $_POST['city_name'];
$email = $_POST['email'];
date_default_timezone_set("Asia/Calcutta");
$insertdate = date("Y-m-d H:i:s");
$sql=$insertdata->update($userid,$first_name,
$last_name,$city_name,$email,$insertdate);
if($sql)
{
echo "Data inserted successfully !";
}
else
{
echo "Data inserted error !";
}
}
?>

```

**For delete data from MySQL first we have to create a table in data base.**

*delete\_process.php:For Delete process in backend*

```

CREATE TABLE `employee` (
  `userid` int(8) NOT NULL AUTO_INCREMENT,
  `first_name` varchar(55) NOT NULL,
  `last_name` varchar(55) NOT NULL,
  `city_name` varchar(55) NOT NULL,
  `email` varchar(50) NOT NULL,
  `datetime` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

*crud.php*

```

<?php
class Crud
{
    private $servername = "localhost";
    private $username = "root";
    private $password = "";

```

```

        private $dbname = "oops_db";
    public $con;
    public $employeeTable = "employee";
        public function __construct()
    {
        try {
            $this->con = new mysqli($this-
>servername, $this->username, $this->password, $this->dbname);
        } catch (Exception $e) {
            echo $e->getMessage();
        }
    }
/* Fetch employee records for show listing */
    public function displayRecord()
    {
        $sql = "SELECT * FROM $this->employeeTable";
        $query = $this->con->query($sql);
        $data = array();
        if ($query->num_rows > 0) {
            while ($row = $query->fetch_assoc()) {
                $data[] = $row;
            }
            return $data;
        }else{
            return false;
        }
    }
    public function deleteRecord($id)
    {
        $sql = "DELETE from $this->employeeTable
where userid=$id";
        $query = $this->con->query($sql);
        $data = array();
        if ($query) {
            return true;
        }else{
            return false;
        }
    }
}
?>
\
```

## view.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Bootstrap Example</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script
    src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"></script>
    <script
    src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</head>
<body>
<?php
include_once("Crud.php");
$viewdata=new Crud();
$customers = $viewdata->displayRecord();
$output = "";
$output .="<table class='table table-striped table-hover'>
    <thead>
        <tr>
            <th>Id</th>
            <th>First Name</th>
            <th>Last Name</th>
            <th>City Name</th>
            <th>Email</th>
            <th>Action</th>
        </tr>
    </thead>
    <tbody>";
foreach ($customers as $customer) {
    $output.= "<tr>
        <td>".$customer['userid']."'</td>
        <td>".$customer['first_name']."'</td>
        <td>".$customer['last_name']."'</td>
        <td>".$customer['city_name']."'</td>
        <td>".$customer['email']."'</td>
```

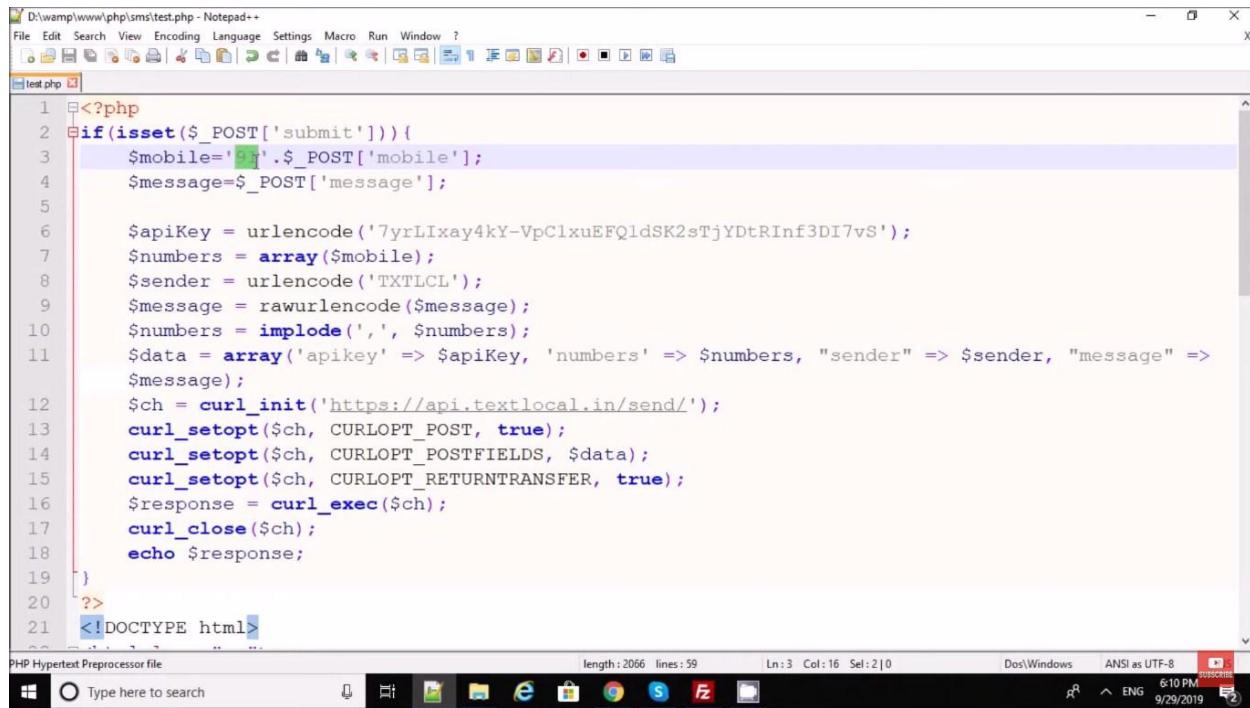
```
        <td><a href='delete_process.php?id='.$customer['userid']."'>Delete</a></td>
    </tr>";
}
$output .= "</tbody>
</table>";
echo $output;
?>
</body>
</html>
```

## *crud.php*

```
<?php

include_once("Crud.php");
$insertdata=new Crud();
$id = $_GET['id'];
$sql=$insertdata->deleteRecord($id);
if($sql)
{
    echo "Data deleted successfully !";
}
else
{
    echo "Error ! Please try again";
}
```

# How to send message or sms in PHP



A screenshot of the Notepad+ application window. The title bar reads "D:\wamp\www\php\sms\test.php - Notepad+". The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Macro, Run, Window, and Help. Below the menu is a toolbar with various icons. The main text area contains PHP code for sending an SMS using cURL. The code includes variables for mobile number, message, and API key, and demonstrates the use of curl\_init, curl\_setopt, curl\_exec, and curl\_close functions. The code ends with an HTML DOCTYPE declaration. The status bar at the bottom shows "PHP Hypertext Preprocessor file", "length : 2066 lines : 59", "Ln : 3 Col : 16 Sel : 2 | 0", "Dos\Windows", "ANSI as UTF-8", and a system tray with icons for battery, network, and date/time.

```
1 <?php
2 if(isset($_POST['submit'])){
3     $mobile='91' . $_POST['mobile'];
4     $message=$_POST['message'];
5
6     $apiKey = urlencode('7yrLIXay4kY-VpClxuEFQ1dSK2sTjYDtRInf3DI7vS');
7     $numbers = array($mobile);
8     $sender = urlencode('TXTLCL');
9     $message = rawurlencode($message);
10    $numbers = implode(',', $numbers);
11    $data = array('apikey' => $apiKey, 'numbers' => $numbers, "sender" => $sender, "message" =>
12    $message);
13    $ch = curl_init('https://api.textlocal.in/send/');
14    curl_setopt($ch, CURLOPT_POST, true);
15    curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
16    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
17    $response = curl_exec($ch);
18    curl_close($ch);
19    echo $response;
20 }
21 ?>
22 <!DOCTYPE html>
```

*index.php*

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Send Message</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">
    <script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.m
in.js"></script>
    <script
    src="https://cdn.jsdelivr.net/npm/popper.js@1.14.3/umd/
popper.min.js"></script>
    <script
    src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstra
p.min.js"></script>
```

```
</head>
<body>

<div class="container">
    <h2>Send Message</h2>
    <p id="message"></p>
        <div class="form-group">
            <label for="email">Sender ID:</label>
            <input type="email" class="form-control" id="senderid"
placeholder="Enter Sender ID" name="senderid">
        </div>
        <div class="form-group">
            <label for="pwd">Mobile No.:</label>
            <input type="text" class="form-control" id="mobile"
placeholder="Enter Mobile No" name="mobile">
        </div>
        <div class="form-group">
            <label for="comment">Message:</label>
            <textarea class="form-control" rows="5" id="message"
name="Message" placeholder="Write Your Message
Here...."></textarea>
        </div>
        <button type="submit" class="btn btn-primary" id="send">Send
Message</button>

</div>
<script>
    $('#send').click(function() {
        $.ajax({
            type: "POST",
            url: "process.php",
            data: {
                senderid:
                    $('#senderid').val(),
                mobile:
                    $('#mobile').val(),
                message:
                    $('#message').val()
            },
            success: function(data) {
                var objJSON =
                    JSON.parse(data);
                $('#message').html(objJSON.message);
            }
        });
    });
</script>
```

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

## process.php

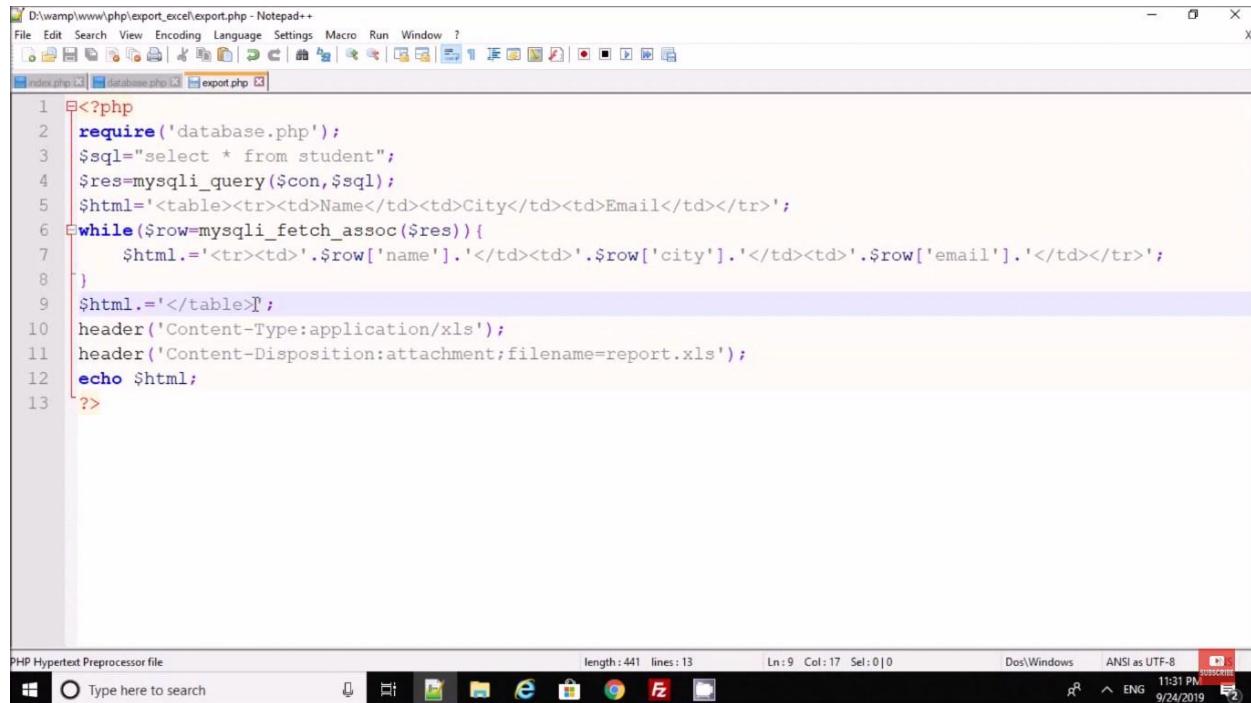
```
<?php
$senderID = $_POST['senderid'];
$mobile = $_POST['mobile'];
$message = $_POST['message'];
/*MESSAGE CODE*/
/* Your authentication key */
$authKey = "234557688009432n";
/* Multiple mobiles numbers separated by comma */
$mobileNumber = $mobile;
/* Sender ID,While using route4 sender id should be 6
characters long. */
$senderId = $senderID;
/* Your message to send, Add URL encoding here. */

$message = urlencode($message);
/* Define route */
$route = "route=4";
/* Prepare you post parameters */
$postData = array(
    'authkey' => $authKey,
    'mobiles' => $mobileNumber,
    'message' => $message,
    'sender' => $senderId,
    'route' => $route
);
/* API URL*/
$url="https://www.fast2sms.com/dev/bulk";
/* init the resource */
$ch = curl_init();
curl_setopt_array($ch, array(
    CURLOPT_URL => $url,
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_POST => true,
    CURLOPT_POSTFIELDS => $postData
    /*,CURLOPT_FOLLOWLOCATION => true */
));
/* Ignore SSL certificate verification */
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0);
curl_setopt($ch, CURLOPT_SSL_VERIFYPeer, 0);
/* get response */
$output = curl_exec($ch);
```

```

/* Print error if any */
if(curl_errno($ch))
{
    echo 'error:' . curl_error($ch);
}
else{
    $array = array(
        "message"      => $output
    );
}
curl_close($ch);
//MESSAGE CODE END
print json_encode($array);
?>

```



The screenshot shows a Notepad++ window with the following PHP code:

```

1 <?php
2 require('database.php');
3 $sql="select * from student";
4 $res=mysqli_query($con,$sql);
5 $html='<table><tr><td>Name</td><td>City</td><td>Email</td></tr>';
6 while($row=mysqli_fetch_assoc($res)){
7     $html.='|  |  |  |
| --- | --- | --- |
| ' . $row['name'] . ' | ' . $row['city'] . ' | ' . $row['email'] . ' |
';
8 }
$html.='</table>';
header('Content-Type:application/xls');
header('Content-Disposition:attachment;filename=report.xls');
echo $html;
?>

```

The code is intended to generate an Excel file by outputting an HTML table structure. It includes file headers and a closing PHP tag.

## PHP oops login

```
CREATE TABLE `users` (
  `id` int(11) NOT NULL,
  `username` varchar(30) NOT NULL AUTO_INCREMENT,
  `password` varchar(30) NOT NULL,
  `fname` varchar(100) NOT NULL,
  PRIMARY KEY(`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

INSERT INTO `users` (`id`, `username`, `password`, `fname`)
VALUES
(1, 'neovic', 'devierte', 'Neovic Devierte'),
(2, 'gemalyn', 'cepe', 'Gemalyn Cepe');
```

### *connection.php*

```
<?php
class DbConnection{

    private $host = 'localhost';
    private $username = 'root';
    private $password = '';
    private $database = 'test';

    protected $connection;

    public function __construct() {
        if (!isset($this->connection)) {

            $this->connection = new mysqli($this->host, $this->username, $this->password, $this->database);

            if (!$this->connection) {
                echo 'Cannot connect to database server';
                exit;
            }
        }

        return $this->connection;
    }
}
?>
```

## *user.php*

```
<?php
include_once('DbConnection.php');

class User extends DbConnection{

    public function __construct() {
        parent::__construct();
    }

    public function check_login($username, $password) {
        $sql = "SELECT * FROM users WHERE username = '$username'
AND password = '$password'";
        $query = $this->connection->query($sql);

        if($query->num_rows > 0){
            $row = $query->fetch_array();
            return $row['id'];
        }
        else{
            return false;
        }
    }

    public function details($sql) {
        $query = $this->connection->query($sql);

        $row = $query->fetch_array();

        return $row;
    }

    public function escape_string($value) {
        return $this->connection->real_escape_string($value);
    }
}
```

## *index.php*

```
<?php

    session_start();

    if(isset($_SESSION['user'])){
        header('location:home.php');
    }
?>
<!DOCTYPE html>
<html>
<head>
    <title>PHP oops login</title>
    <link
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
<div class="container">
    <h1 class="page-header text-center">PHP oops login</h1>
    <div class="row">
        <div class="col-md-4 col-md-offset-4">
            <div class="login-panel panel panel-primary">
                <div class="panel-heading">
                    <h3 class="panel-title"><span
class="glyphicon glyphicon-lock"></span> Login
                </h3>
                </div>
                <div class="panel-body">
                    <form method="POST" action="login.php">
                        <fieldset>
                            <div class="form-group">
                                <input class="form-
control" placeholder="Username" type="text" name="username"
autofocus required>
                            </div>
                            <div class="form-group">
                                <input class="form-
control" placeholder="Password" type="password" name="password"
required>
                            </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</div>
```

```

                                <button type="submit"
name="login" class="btn btn-lg btn-primary btn-block"><span
class="glyphicon glyphicon-log-in"></span> Login</button>
                            </fieldset>
                        </form>
                    </div>
                </div>
                <?php
                    if(isset($_SESSION['message'])) {
                        ?>
                        <div class="alert alert-info
text-center">
                            <?php echo
$_SESSION['message']; ?>
                            </div>
                        <?php
                            unset($_SESSION['message']);
                        }
                    ?>
                </div>
            </div>
        </body>
    </html>

```

## *login.php*

```

<?php

session_start();

include_once('User.php');

$user = new User();

if(isset($_POST['login'])) {
    $username = $user->escape_string($_POST['username']);
    $password = $user->escape_string($_POST['password']);

    $auth = $user->check_login($username, $password);

```

```

        if (!$auth) {
            $_SESSION['message'] = 'Invalid username or
password';
            header('location:index.php');
        }
        else{
            $_SESSION['user'] = $auth;
            header('location:home.php');
        }
    }
else{
    $_SESSION['message'] = 'You need to login first';
    header('location:index.php');
}
?>

```

## *home.php*

```

<?php
session_start();

if (!isset($_SESSION['user']) || (trim($_SESSION['user']) ==
'')) {
    header('location:index.php');
}

include_once('User.php');

$user = new User();

$sql = "SELECT * FROM users WHERE id = '". $_SESSION['user']."' ";
$row = $user->details($sql);

?>
<!DOCTYPE html>
<html>
<head>
    <title>PHP OOPS Login</title>
    <link
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" rel="stylesheet">
</head>
<body>

```

```

<div class="container">
    <h1 class="page-header text-center">PHP OOPS Login</h1>
    <div class="row">
        <div class="col-md-4 col-md-offset-4">
            <h2>Welcome to Homepage </h2>
            <h4>User Info: </h4>
            <p>Name: <?php echo $row['fname']; ?></p>
            <p>Username: <?php echo $row['username']; ?>
        </p>
        <p>Password: <?php echo $row['password']; ?>
    </p>
    <a href="logout.php" class="btn btn-danger"><span class="glyphicon glyphicon-log-out"></span> Logout</a>
</div>
</body>
</html>

```

*logout.php*

```

<?php
    session_start();
    session_destroy();

    header('location:index.php');
?>

```

## PHP oops login

```

CREATE TABLE `tblusers` (
    `id` int(11) NOT NULL,
    `FullName` varchar(120) DEFAULT NULL,
    `Username` varchar(120) DEFAULT NULL,
    `UserEmail` varchar(200) DEFAULT NULL,
    `Password` varchar(250) DEFAULT NULL,
    `RegDate` timestamp NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

## function.php

```
<?php
define('DB_SERVER','localhost');
define('DB_USER','root');
define('DB_PASS','');
define('DB_NAME', 'userdb');
class DB_con
{
    function __construct()
    {
        $con = mysqli_connect(DB_SERVER,DB_USER,DB_PASS,DB_NAME);
        $this->dbh=$con;

        if (mysqli_connect_errno())
        {
            echo "Failed to connect to MySQL: " . mysqli_connect_error();
        }
    }

    public function usernameavailblty($uname) {
        $result =mysqli_query($this->dbh,"SELECT Username FROM tblusers
        WHERE Username='$uname'");
        return $result;
    }

    public function registration($fname,$uname,$uemail,$password)
    {
        $ret=mysqli_query($this->dbh,"insert into
        tblusers(FullName,Username,UserEmail,Password)
        values('$fname','$uname','$uemail','$password')");
        return $ret;
    }

    public function signin($uname,$pasword)
    {
        $result=mysqli_query($this->dbh,"select id,FullName from
        tblusers where Username='$uname' and Password='$pasword'");
        return $result;
    }

?>
```

Like index.php may be

```
<?php

include_once('function.php');

$userdata=new DB_con();
if(isset($_POST['submit']))
{

$fname=$_POST['fullname'];
$uname=$_POST['username'];
$uemail=$_POST['email'];
$password=md5($_POST['password']);

$sql=$userdata->registration($fname,$uname,$uemail,$password);
if($sql)
{

echo "<script>alert('Registration successfull.');//</script>";
echo "<script>window.location.href='signin.php'</script>";
}
else
{

echo "<script>alert('Something went wrong. Please try
again');//</script>";
echo "<script>window.location.href='signin.php'</script>";
}
}

?>
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<!-- This file has been downloaded from Bootsnipp.com. Enjoy! -->
<title>User Registration using PHP OOPS Concept</title>
<meta name="viewport" content="width=device-width, initial-
scale=1">
<link href="assests/style.css" rel="stylesheet">
<script src="assests/jquery-1.11.1.min.js"></script>
<script src="assests/bootstrap.min.js"></script>
<script>
function checkusername(va) {
```

```

$.ajax({
type: "POST",
url: "check_availability.php",
data:'username='+va,
success: function(data) {
$("#usernameavailblty").html(data);
}
} );
}

</script>
</head>
<body>
<form class="form-horizontal" action=' ' method="POST">
<fieldset>
<div id="legend" align="center">
<legend class="">User Registration using PHP OOPS
Concept</legend>
</div>

<div class="control-group">
<!-- Fullname -->
<label class="control-label" for="username">Fullname</label>
<div class="controls">
<input type="text" id="username" name="fullname" placeholder="" class="input-xlarge" required="true">
</div>
</div>

<div class="control-group">
<!-- Username -->
<label class="control-label" for="username">Username</label>
<div class="controls">
<input type="text" id="username" name="username" onblur="checkusername(this.value)" class="input-xlarge" required="true">
<span id="usernameavailblty"></span>
</div>
</div>

<div class="control-group">
<!-- E-mail -->
<label class="control-label" for="email">E-mail</label>
<div class="controls">
<input type="email" id="email" name="email" placeholder="" class="input-xlarge" required="true">

```

```

</div>
</div>

<div class="control-group">
<!-- Password-->
<label class="control-label" for="password">Password</label>
<div class="controls">
<input type="password" id="password" name="password"
placeholder="" class="input-xlarge" required="true">
</div>
</div>

<div class="control-group">
<!-- Button -->
<div class="controls">
<button class="btn btn-success" type="submit" id="submit"
name="submit">Register</button>
</div>
</div>

<div class="control-group">
<div class="controls">
Already registered <a href="#">Signin</a>
</div>
</div>

</fieldset>
</form>
<script >
</script>
</body>
</html>

```

### *check\_availability.php*

```

<?php

include_once('function.php');

$uusername=new DB_con();

$uname= $_POST["username"];

$sql=$uusername->usernameavailblty($uname);
$num=mysqli_num_rows($sql);
if($num > 0)

```

```

{
echo "<span style='color:red'> Username already associated with
another account .</span>";
echo "<script>$('#submit').prop('disabled',true);</script>";
} else{

echo "<span style='color:green'> Username available for
Registration .</span>";
echo "<script>$('#submit').prop('disabled',false);</script>";
} ?>

```

## *signin.php*

```

<?php

include_once('function.php');

$userdata=new DB_con();
if(isset($_POST['submit']))
{

$fname=$_POST['fullname'];
$uname=$_POST['username'];
$uemail=$_POST['email'];
$password=md5($_POST['password']);

$sql=$userdata->registration($fname,$uname,$uemail,$password);
if($sql)
{

echo "<script>alert('Registration successfull.');//</script>";
echo "<script>window.location.href='signin.php'</script>";
}
else
{

echo "<script>alert('Something went wrong. Please try
again');//</script>";
echo "<script>window.location.href='signin.php'</script>";
}
}

?>
<!DOCTYPE html>
<html lang="en">
<head>
```

```
<meta charset="utf-8">
<title>User Registration using PHP OOPs Concept</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link href="assests/style.css" rel="stylesheet">
<script src="assests/jquery-1.11.1.min.js"></script>
<script src="assests/bootstrap.min.js"></script>
<script>
function checkusername(va) {
$.ajax({
type: "POST",
url: "check_availability.php",
data:'username='+va,
success: function(data) {
$("#usernameavailblty").html(data);
}
});
}

</script>
</head>
<body>
<form class="form-horizontal" action=' ' method="POST">
<fieldset>
<div id="legend" align="center">
<legend class="">User Registration using PHP OOPs
Concept</legend>
</div>

<div class="control-group">
<!-- Fullname -->
<label class="control-label" for="username">Fullname</label>
<div class="controls">
<input type="text" id="username" name="fullname" placeholder=""
class="input-xlarge" required="true">
</div>
</div>

<div class="control-group">
<!-- Username -->
<label class="control-label" for="username">Username</label>
<div class="controls">
<input type="text" id="username" name="username"
onblur="checkusername(this.value)" class="input-xlarge"
required="true">
<span id="usernameavailblty"></span>
</div>
```

```

</div>

<div class="control-group">
<!-- E-mail -->
<label class="control-label" for="email">E-mail</label>
<div class="controls">
<input type="email" id="email" name="email" placeholder="" class="input-xlarge" required="true">
</div>
</div>

<div class="control-group">
<!-- Password-->
<label class="control-label" for="password">Password</label>
<div class="controls">
<input type="password" id="password" name="password" placeholder="" class="input-xlarge" required="true">
</div>
</div>

<div class="control-group">
<!-- Button -->
<div class="controls">
<button class="btn btn-success" type="submit" id="submit" name="submit">Register</button>
</div>
</div>

<div class="control-group">
<div class="controls">
Already registered <a href="#">Signin</a>
</div>
</div>

</fieldset>
</form>
<script >
</script>
</body>
</html>

```

## *welcome.php*

```

<?php
session_start();

```

```

if(strlen($_SESSION['uid'])=="")
{
header('location:logout.php');
} else {
?><!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>User Registration using PHP OOPS Concept</title>
<meta name="viewport" content="width=device-width, initial-
scale=1">
<link href="assests/style.css" rel="stylesheet">
<script src="assests/jquery-1.11.1.min.js"></script>
<script src="assests/bootstrap.min.js"></script>
</head>
<body>
<form class="form-horizontal" action=' ' method="POST">
<fieldset>
<div id="legend">
<legend class="" align="center">Welcome Back : <?php echo
$_SESSION['fname'];?></legend>
</div>

<div class="control-group" align="center">
<!-- Button -->
<div class="controls">
<a href="logout.php" class="btn btn-success" type="submit"
name="signin">Logout</a>
</div>
</div>

</fieldset>
</form>
<script >
</script>
</body>
</html>
<?php } ?>

```

## *logout.php*

```

<?php
session_start();
session_destroy();
header('location:signin.php');
?>

```

## PHP Caching

```
<?php
//Caching is a temporary storage location, to access data more quickly. Caching is used where the
original data is expensive to fetch.
$start=microtime(true);
$con=new PDO("mysql:host=localhost;dbname=youtube","root","");
$sql="select student.name, city.city, game.game, study.study, teacher.teacher from
student,city,game,study,teacher,fee where student.city=city.id and student.game=game.id and
student.study=study.id and student.teacher=teacher.id and student.id=fee.student_id";
$stmt=$con->prepare($sql);
$stmt->execute();
$arr=$stmt->fetchAll(PDO::FETCH_ASSOC);
$str=<table border='1'>;
$str.=<tr><td>Name</td><td>City</td><td>Game</td><td>Study</td><td>Teacher</td></tr>;
foreach($arr as $list){
    $str.=<tr><td>".$list['name']."'</td><td>".$list['city']."'</td><td>".$list['game']."'</td><td>".$list['study']."'</td><td>".$list['teacher']."'</td></tr>;
}
$str.=</table>;
echo $str;
$end=microtime(true);
echo round($end-$start,4);
?>
```

```
?php
//Caching is a temporary storage location, to access data more quickly. Caching is used where the original data is
expensive to fetch.
$start=microtime(true);
$con=new PDO("mysql:host=localhost;dbname=youtube","root","");

$cache_file="cache/index.cache.php";
if(file_exists($cache_file) && filemtime($cache_file) > time()-20){
    echo "From Cache<br/>";
    include($cache_file);
} else{
    $sql="select student.name, city.city, game.game, study.study, teacher.teacher from student,city,game,study,
teacher,fee where student.city=city.id and student.game=game.id and student.study=study.id and student.
teacher=teacher.id and student.id=fee.student_id";
    $stmt=$con->prepare($sql);
    $stmt->execute();
    $arr=$stmt->fetchAll(PDO::FETCH_ASSOC);
    $str=<table border='1'>;
    $str.=<tr><td>Name</td><td>City</td><td>Game</td><td>Study</td><td>Teacher</td></tr>;
    foreach($arr as $list){
        $str.=<tr><td>".$list['name']."'</td><td>".$list['city']."'</td><td>".$list['game']."'</td><td>".$list
['study']."'</td><td>".$list['teacher']."'</td></tr>;
    }
    $str.=</table>;
    $handle=fopen($cache_file,'w');
    fwrite($handle,$str);
    fclose($handle);
    echo "Cache Created<br/>";
    echo $str;
}
$end=microtime(true);
echo round($end-$start,4);
?>
```

```

<?php
//Caching is a temporary storage location, to access data more quickly. Caching
is used where the original data is expensive to fetch.
$start=microtime(true);
$con=new PDO("mysql:host=localhost;dbname=youtube","root","");
$cache_file="cache/index.cache.php";
if(file_exists($cache_file) && filemtime($cache_file) > time()-20){
    echo "From Cache<br/>";
    include($cache_file);
}else{
    $sql="select student.name, city.city, game.game, study.study, teacher.teacher
from student,city,game,study,teacher,fee where student.city=city.id and
student.game=game.id and student.study=study.id and student.teacher=teacher.id
and student.id=fee.student_id";
    $stmt=$con->prepare($sql);
    $stmt->execute();
    $arr=$stmt->fetchAll(PDO::FETCH_ASSOC);
    $str=<table border='1'>;
    $str.=<tr><td>Name</td><td>City</td><td>Game</td><td>Study</td><td>Teacher</td></tr>;
    foreach($arr as $list){
        $str.=<tr><td>".$list['name']."'</td><td>".$list['city']."'</td><td>".$list['game']."'</td><td>".$list['study']."'</td><td>".$list['teacher']."'</td></tr>;
    }
    $str.=</table>;
    $handle=fopen($cache_file,'w');
    fwrite($handle,$str);
    fclose($handle);
    echo "Cache Created<br/>";
    echo $str;
}
$end=microtime(true);
echo round($end-$start,4);
?>

```

## Array



### Multidimensional Associative Array :

Name	Physics	Chemistry	Math
Krishana	85	89	78
Salman	68	79	73
Mohan	62	92	67

```
$marks = array (
    "Krishana" => array("physics" => 85, "chemistry" => 89, "math" => 78),
    "Salman" => array("physics" => 68, "chemistry" => 79, "math" => 73),
    "Mohan" => array("physics" => 62, "chemistry" => 92, "math" => 67)
);

$marks = [
    "Krishana" => [
        "physics" => 85,
        "maths" => 78,
        "chemistry" => 89
    ],
    "Salman" => [
        "physics" => 68,
        "maths" => 73,
        "chemistry" => 79
    ],
    "Mohan" => [
        "physics" => 62,
        "maths" => 67,
        "chemistry" => 92
    ]
];
```

```
<?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 />";
}
?>
```

Loop through and print all the values of an associative array:

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");

foreach($age as $x=>$x_value)
{
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>
```

```

1         "chemistry" => 79
2     ],
3     "Mohan" => [
4         "physics" => 62,
5         "maths" => 67,
6         "chemistry" => 92
7     ]
8 ];
9
10    foreach($marks as $key => $v1){
11        echo $key ;
12        foreach($v1 as $v2){
13            echo $v2 . " ";
14        }
15        echo "<br>";
16    }
17
18
19
20
21         "chemistry" => 92
22     ];
23 ];
24 echo "<table border='2px' cellpadding='5px'>";
25 foreach($marks as $key => $v1){
26     echo "<tr>
27         <td>$key</td>" ;
28     foreach($v1 as $v2){
29         echo "<td> $v2 </td>";
30     }
31     echo "</tr>";
32 }
33 echo "</table>";

```

Krishana85 78 89  
 Salman68 73 79  
 Mohan62 67 92

Array  
 ( [Krishana] => Array  
 ( [physics] => 85  
 [maths] => 78  
 [chemistry] => 89 )  
 [Salman] => Array  
 ( [physics] => 68  
 [maths] => 73  
 [chemistry] => 79 )  
 [Mohan] => Array  
 ( [physics] => 62  
 [maths] => 67  
 [chemistry] => 92 ) )

Krishana	85	78	89
Salman	68	73	79
Mohan	62	67	92

Array  
 ( [Krishana] => Array  
 ( [physics] => 85  
 [maths] => 78  
 [chemistry] => 89 )  
 [Salman] => Array  
 ( [physics] => 68  
 [maths] => 73  
 [chemistry] => 79 ) )

## PHP Array Functions

- **count()** - returns the number of elements in an array.
- **in\_array()** - checks if a value exists in an array.
- **array\_push()** - adds one or more elements to the end of an array.
- **array\_pop()** - removes and returns the last element of an array.
- **array\_shift()** - removes and returns the first element of an array.
- **array\_unshift()** - adds one or more elements to the beginning of an array.
- **array\_reverse()** - reverses the order of the elements in an array.
- **array\_merge()** - merges two or more arrays into one array.
- **array\_slice()** - extracts a slice of an array.
- **array\_splice()** - removes and replaces elements from an array.
- **array\_keys()** - returns an array of the keys of an array.
- **array\_values()** - returns an array of the values of an array.
- **array\_flip()** - flips the keys and values of an array.
- **array\_unique()** - removes duplicate values from an array.
- **array\_search()** - searches for a value in an array and returns its key.
- **array\_rand()** - returns one or more random keys from an array.
- **array\_walk()** - applies a user-defined function to each element of an array.
- **array\_map()** - applies a user-defined function to each element of an array and returns a new array with the results.
- **array\_filter()** - filters an array using a user-defined function and returns a new array with the filtered values.
- **array\_reduce()** - reduces an array to a single value using a user-defined function.
- **array\_column()** - returns an array of values from a single column of a multi-dimensional array.
- **array\_intersect()** - returns an array of the values that exist in two or more arrays.
- **array\_diff()** - returns an array of the values that exist in one array but not in another.



## PHP Array : Extract() Function

```
$color = array('a' => 'red', 'b' => 'green', 'c' => 'blue');
```

↓      ↓      ↓  
\$a      \$b      \$c

**extract(array, extract\_rules, prefix)**



Yahoo Baba

www.yahooomba.net

Yahoo  
Baba



## PHP Array : Compact Function

```
$first = "red";                          Array(  
$second = "green";      →             "first" => "red",  
$third = "blue";                         "second" => "green",  
                                              "third" => "blue",  
                                              )  
                                              )
```

**compact(var1, var2, var3.....)**



Yahoo Baba

www.yahooomba.net

Yahoo  
Baba



## PHP Array : Range() Function

1      5

array(1, 2, 3, 4 ,5)

**range(start, end, step)**



Yahoo Baba

www.yahooomba.net

Yahoo  
Baba

```
<?php  
  
// $food = array('orange', 'banana', 'apple', '55', 'grapes');  
$food = array('a' => 'orange', 'b' => 'banana', 'c' => 'apple', 'd' => 'grapes'  
  
echo array_search('apple', $food);  
  
?>
```



## PHP Array – Array\_Map() Function :

```
$a = array(  
    "Bill" => 10,  
    "Joe" => 20,  
    "Peter" => 30  
);
```

function myFunction(){  
}

Red arrows point from the values in the array to the function definition, indicating they are passed as arguments.

**array\_map(function , array)**

```
$a = array(  
    "Bill" => 10,  
    "Joe" => 20,  
    "Peter" => 30  
);
```

function myFunction(){  
}

A blue bracket labeled "Return Array" points back to the original array variable \$a, indicating the result of the function call.

**array\_map(function , array , array2 , array3 ....)**

```
<?php
function square($n){
    return $n * $n;
}

$a = [1, 2, 3, 4, 5];

$newArray = array_map('square', $a);

echo "<pre>"; | ①
print_r($newArray);
echo "</pre>";
```



## PHP Array – Merge & Combine Functions :

```
$a = ["Sanjay", "Aman", "Rehman"];
$b = ["Karan"];
["Sanjay", "Aman", "Rehman", "Karan"]
```

- `array_merge()` Index or Associative Array
- `array_merge_recursive()` Multidimensional Associative Array
- `array_combine()` Index Array

```
<?php
$fruit = ["orange", "banana", "grapes"];

$veggie = ['carrot', 'pea'];

$color = ['red', 'blue'];

$newArray = array_merge($fruit, $veggie, $color);

echo "<pre>";
print_r($newArray);
echo "</pre>";
```

```

<?php
$fruit = ['a' => "orange", 'b' => "banana", 'c' => "grapes"];

$veggie = ['carrot', 'pea'];

$newArray = array_merge($fruit,$veggie);

echo "<pre>";
print_r($newArray);
echo "</pre>";

```

```

<?php
$name = array("Ram","Mohan","Salman");

$age = array("35","37","43");

$newArray = array_combine($name,$age);

echo "<pre>";
print_r($newArray);

```

## PHP Array\_fill\_keys() Function :

```
$a = ["a", "b", "c", "d"];
```

```

Array(
    "a" => "test",
    "b" => "test",
    "c" => "test",
    "d" => "test"
);
```

```
array_fill_keys(array , value)
```



## PHP Array – Array\_Reduce() Function :

```
$a = array(  
    10, → function myFunction(){  
    20, → }  
    30 →  
);
```

Return Single Value ←

**array\_reduce(array , function , initial)**

```
<?php  
$a1 = array("a"=>"red", "b"=>"green", "c"=>"blue", "d"=>"yellow");  
  
$a2 = array("a"=>"red", "f"=>"green", "d"=>"purple");  
  
$newArray = array_intersect($a1, $a2);  
  
echo "<pre>";  
print_r($newArray);  
echo "</pre>";  
<?php  
$values=['Mohammad','','10,0,false,"2020",null,true];  
$filtered = array_filter($values);  
echo "<pre>";  
print_r($values);  
print_r($filtered);
```



## PHP Array\_fill() Function :

```
Array(  
    [0] => "test",  
    [1] => "test",  
    [2] => "test",  
    [3] => "test"  
);
```

```
Array(  
    [3] => "test",  
    [4] => "test",  
    [5] => "test"  
);
```

**array\_fill(index , number , value)**

## PHP Array : Difference Functions

- array\_diff
- array\_diff\_key
- array\_diff\_assoc
- array\_diff\_uassoc
- array\_diff\_ukey
- array\_udiff\_assoc
- array\_udiff\_uassoc
- array\_udiff



## PHP Array : Difference Functions

### Different Values

```
$a = ["a" => "Sanjay", "b" => "Aman", "c" => "Mohan"];
```

```
$b = ["a" => "Sanjay", "d" => "Mohan"];
```

▶

**array\_diff**



## PHP Array : Intersect Functions

### Match Keys

```
$a = ["a" => "Sanjay", "b" => "Aman", "c" => "Mohan"];  
↓  
$b = ["a" => "Sanjay", "d" => "Mohan"];
```

array\_intersect\_key



## PHP Array : Difference Functions

### Different Keys

```
$a = ["a" => "Sanjay", b => "Aman", c => "Mohan"];  
↑  
$b = ["a" => "Sanjay", "d" => "Mohan"];
```

array\_diff\_key



## PHP Array : Intersect Functions

### Match Values

```
$a = ["a" => "Sanjay", "b" => "Aman", "c" => "Mohan"];  
↑  
$b = ["a" => "Sanjay", "d" => "Mohan"];
```

array\_intersect

```
<?php  
$a1 = array("a"=>"red","b"=>"green","c"=>"blue","d"=>"yellow");  
  
$a2 = array("a"=>"red","f"=>"green","d"=>"purple");  
  
$newArray = array_intersect($a1, $a2);  
  
echo "<pre>";  
print_r($newArray);  
echo "</pre>";  
  
<?php  
$values=['Mohammad','','10,0',false,"2020",null,true];  
$filtered = array_filter($values);  
echo "<pre>";  
print_r($values);  
print_r($filtered);
```



## PHP Array – Array\_Reduce() Function :

```
$a = array(  
    10, → function myFunction(){  
    20, →  
    30 }  
);  
                    Return Single Value ←
```

array\_reduce(array , function , initial)

```

<?php
function myFunction($n , $m){
    return $n . "-" . $m;
}

$a = ['orange', 'banana', 'apple'];

$newArray = array_reduce($a, 'myFunction', 'lemon');

echo "<pre>";
print_r($newArray);

```

```

$array = array(
    array(
        'id' => 2201,
        'first_name' => 'Anil',
        'last_name' => 'Kapoor',
    ),
    array(
        'id' => 2202,
        'first_name' => 'Salman',
        'last_name' => 'Khan',
    ),
    array(
        'id' => 2203,
        'first_name' => 'John',
        'last_name' => 'Abraham',
    )
);

```

## array\_column

```

Array
(
    [0] => Anil
    [1] => Salman
    [2] => John
)

```

\$a = ["Red", "Green", "Blue", "Orange", "Brown"];

```

Array
(
    [0] => Array(
        [0] => Red
        [1] => Green
    ),
    [1] => Array(
        [2] => Blue
        [3] => Orange
    ),
    [2] => Array (
        [4] => Brown
    )
)

```

## array\_chunk (array , size)

## String Function

```
<?php

$data = "12344 $ # % @ ! ^ ' ". 'ram' . " ' & ' ' Have a nice day &
best of luck";

echo "<br/><br/><a href=urlencode0.php?data=".urlencode($data)."
target='_blank'> click here for send data with urlencode()
</a>";

echo "<br/><br/><a href=urlencode0.php?data=".$data."
target='_blank'> click here for send data without
urlencode() </a>";
```

```
?>
```

```
urlencode($string);
```

- Letters, numbers, underscores, dashes are unchanged
- Reserved characters become % + 2 digit hexadecimal value
- Spaces become + sign

```
<html>
  <head>
    <title>First Page</title>
  </head>
  <body>
    <?php $link_name = "Second Page"; ?>
    <?php $id = 2; ?>
    <?php $company = "Johnson & Johnson"; ?>
<a href="second_page.php?id=<?php echo $id; ?>&company=<?php echo urlencode($company); ?>">
-<?php echo $link_name; ?></a>
  </body>
</html>
```

1. `strlen()`
2. `str_replace()`
3. `strstr()`
4. `substr()`
5. `strtolower()`
6. `strtoupper()`
7. `str_word_count()`
8. `strrev()`
9. `strpos()`
10. `ucwords()`
11. `ucfirst()`
12. `lcfirst()`
13. `wordwrap()`
14. `trim()`
15. `strip_tags()`

## PHP String : htmlentities Function





## PHP String : htmlentities Function

`htmlentities(string, flags)`



- **ENT\_COMPAT** Default. Encodes only double quotes
- **ENT\_QUOTES** Encodes double and single quotes
- **ENT\_NOQUOTES** Does not encode any quotes

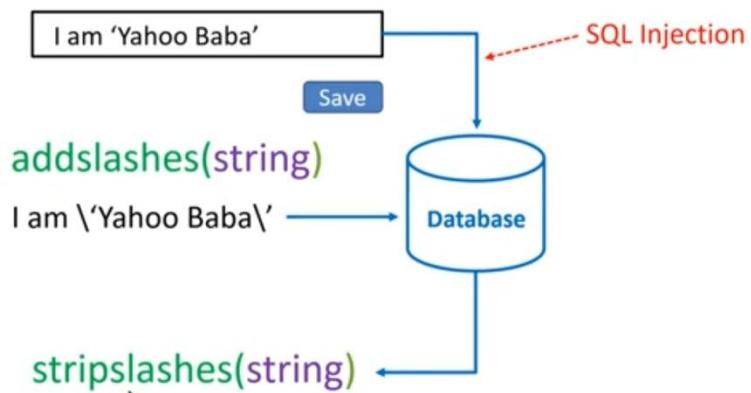
`htmlspecialchars(string, flags)`



`htmlspecialchars_decode(string, flags)`



## PHP String : addslashes & stripslashes Functions





## PHP String Length & Count Functions

`$a = "Hello World";` → 11  
`strlen(string)`

`$a = "Hello World";` → 2  
`str_word_count(string, return)`

`$a = "Hello World";` → World  
`substr_count(string, substring, start, length)`



## PHP String : Find Position Functions

`$a = "I love php, I love php too!";`

`strpos(string, find, start)` → 7

`strrpos(string, find, start)` → 19

case-sensitive      `php`    `Php`



## PHP String : Compare Functions

`hello`                `Hello`

`strcmp(string1, string2)`                case-sensitive  
`strncmp(string1, string2, length)`

`strcasecmp(string1, string2)`                case-insensitive  
`strncasecmp(string1, string2 , length)`





## PHP String : Search String Functions

`$a = "I love php, I love php too!";`

`strstr(string, search, before_search) → , I love php too!`

`strchr(string, search, before_search)`

`strrchr(string, search) → too!`

case-sensitive

php PHP



Yahoo Baba

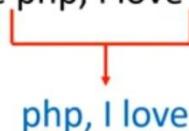
www.yahooomba.net

Yahoo  
Baba



## PHP String : Substr() Function

`$a = "I love php, I love php too!";`



`substr(string, start, length)`



Yahoo Baba

www.yahooomba.net

Yahoo  
Baba



## PHP String : Str\_replace() Function

`$a = "I love php, I love php too!";`

`I love python, I love python too!`

`str_replace(find, replace, string)` case-sensitive  
PHP

`str_ireplace(find, replace, string)`



Yahoo Baba

www.yahooomba.net

Yahoo  
Baba



## PHP String Reverse Function

hello → olleh

`strrev(string)`



www.yahooomba.net

Yahoo  
Baba



## PHP String Shuffle Function

hello → llohe



eohll

lheloo

`str_shuffle(string)`

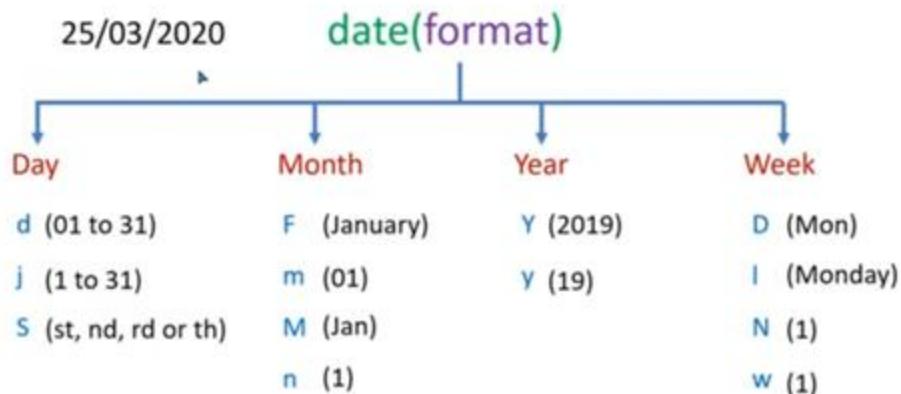


www.yahooomba.net

Yahoo  
Baba



## PHP String : date() Function



www.yahooomba.net



## PHP String : date() Function

