PHP Tutorial - Detailed Guide

Getting Up and Running with PHP

To begin using PHP, you need a server environment. This can be a local server setup like XAMPP, WAMP, or MAMP, which includes Apache, MySQL, and PHP. These tools let you run PHP scripts on your local machine for testing purposes.

Example:

Install XAMPP and place your PHP files in the 'htdocs' folder. Access them via http://localhost/yourfile.php

Your First PHP Script

A PHP script starts with the <?php tag and ends with ?>. Inside, you can write commands like echo or functions.

Example:

```
<?php
  echo 'Hello, PHP world!';
?>
```

Installing PHP

You can install PHP individually from the official PHP website, or as part of packages like XAMPP or WAMP. Installation allows you to run PHP scripts and configure settings through php.ini.

Example:

Download and install PHP from https://www.php.net/downloads.php and run php -v in terminal to verify.

Other Ways to Run PHP

Besides a web server, PHP scripts can run from the command line or using Docker containers for isolated environments.

Example:

php script.php

(This runs a PHP script named script.php from the terminal.)

Creating Your First Script

Create a file with a .php extension and write your PHP code inside. Then access it via a web server.

Example:

File: hello.php

```
<?php
echo 'Welcome to PHP!';
?>
```

Learning the Language

PHP is a scripting language used mainly for web development. It supports imperative, object-oriented, and functional programming styles.

Example:

Variables do not need explicit types:

```
<?php
$name = 'John';
echo $name;
?>
```

PHP Language Basics

PHP uses semicolons to end statements and has C-style control structures like if, while, and for.

Example:

```
<?php
for ($i = 0; $i < 5; $i++) {
  echo $i;
}
?>
```

Using Variables in PHP

Variables start with a $ sign and store different types of data. PHP is loosely typed.

Example:

```
<?php
$age = 25;
echo $age;
?>
```

Understanding Data Types

PHP has several data types: integers, floats, strings, arrays, objects, and more.

Example:

```
<?php
$number = 10;
$price = 19.99;
$text = 'PHP';
```

?>

Operators and Expressions

PHP supports arithmetic, assignment, comparison, and logical operators.

Example:

<?php

$a = 5;

$b = 10;

echo $a + $b;

?>

Strings

Strings can be created using single or double quotes. Double quotes allow variable interpolation.

Example:

<?php

$name = 'Alice';

echo "Hello, $name";

?>

Arrays

Arrays hold multiple values. PHP supports indexed and associative arrays.

Example:

<?php

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

```
echo $colors[0];
?>
```

Functions

Functions let you encapsulate code blocks for reuse. They can take parameters and return values.

Example:

```
<?php
function greet($name) {
  return "Hello, $name!";
}
echo greet('Bob');
?>
```

Objects

PHP supports object-oriented programming with classes and objects.

Example:

```
<?php
class Car {
  public $color = 'red';
}
$car = new Car();
echo $car->color;
?>
```

Using PHP in Practice

PHP is used to create dynamic websites, manage forms, sessions, databases, and more.

Example:

Use PHP to display the current date:

```
<?php
echo date('Y-m-d');
?>
```

Handling HTML Forms with PHP

PHP retrieves form data via $_GET or $_POST depending on the form method.

Example:

```
<form method='post'><input name='name'><input type='submit'></form>
<?php
echo $_POST['name'];
?>
```

Preserving State With Query Strings, Cookies, and Sessions

PHP uses sessions, cookies, and query strings to maintain state.

Example:

```
<?php
session_start();
$_SESSION['user'] = 'admin';
echo $_SESSION['user'];
?>
```

Working with Files and Directories

PHP provides functions like fopen(), fread(), fwrite() to manage files.

Example:

```
<?php
file_put_contents('log.txt', 'Log entry');
?>
```

Introducing Databases and SQL

PHP connects to databases like MySQL using mysqli or PDO. SQL is used to query the database.

Example:

```
<?php
$conn = new mysqli('localhost', 'user', 'pass', 'db');
?>
```

Retrieving Data from MySQL with PHP

Use SELECT statements with mysqli or PDO to fetch data.

Example:

```
<?php
$result = $conn->query('SELECT * FROM users');
while($row = $result->fetch_assoc()) {
echo $row['name'];
}
?>
```

Manipulating MySQL Data with PHP

Use INSERT, UPDATE, DELETE to modify data.

Example:

```
<?php
$conn->query("INSERT INTO users (name) VALUES ('Alice')");
?>
```

Making Your Job Easier with PEAR

PEAR is a repository of PHP components. Install it to reuse existing libraries.

Example:

```
pear install HTTP_Request2
```

PHP and the Outside World

PHP can send requests to APIs, access files, or run shell commands.

Example:

```
<?php
$data = file_get_contents('https://api.example.com');
echo $data;
?>
```

Generating Images with PHP

Use the GD library to create dynamic images.

Example:

```
<?php
$img = imagecreatetruecolor(100, 100);
$bg = imagecolorallocate($img, 255, 0, 0);
imagefill($img, 0, 0, $bg);
imagepng($img);
?>
```

Working with XML

PHP can read and write XML using SimpleXML and DOM extensions.

Example:

```
<?php
$xml = simplexml_load_string('<note><to>User</to></note>');
echo $xml->to;
?>
```

Writing High-Quality Code

Use indentation, comments, and naming conventions. Validate and test your code.

Example:

```
<?php
// This function adds two numbers
function add($a, $b) {
  return $a + $b;
}
?>
```