Chapter 2: PHP Basics

2.1 Variables and Data Types

Variables in PHP are used to store data, and they start with a $ sign followed by the variable name. PHP supports several data types:

String: A sequence of characters, e.g., "Hello, World!"

Integer: Whole numbers, e.g., 123

Float: Numbers with decimal points, e.g., 3.14

Boolean: Represents TRUE or FALSE

Array: A collection of values, e.g., [1, 2, 3, 4, 5]

Object: Instances of classes

NULL: A special type that represents a variable with no value

Example of variable usage:

```
<?php
  $string = "Hello, World!";
  $integer = 123;
  $float = 3.14;
  $boolean = true;
  $array = array(1, 2, 3, 4, 5);
```

```
  echo $string; // Outputs: Hello, World!
  echo $integer; // Outputs: 123
  echo $float; // Outputs: 3.14
  echo $boolean; // Outputs: 1 (true is represented as 1)
  print_r($array); // Outputs: Array ( [0] => 1 [1] => 2 [2] => 3 [3] => 4 [4] => 5 )
?>
```

2.2 Operators and Expressions

Operators are used to perform operations on variables and values. Here are some common operators:

Arithmetic Operators: +, -, *, /, %

Assignment Operators: =, +=, -=, *=, /=

Comparison Operators: ==, !=, >, <, >=, <=

Logical Operators: && (AND), || (OR), ! (NOT)

Example:

```
<?php
  $a = 10;
  $b = 5;

  // Arithmetic
  echo $a + $b; // Outputs: 15
```

```
// Comparison
if ($a > $b) {
  echo "a is greater than b";
}

// Logical
if ($a > 5 && $b < 10) {
  echo "Both conditions are true";
}
?>
```

2.3 Conditional Statements

Conditional statements are used to perform different actions based on different conditions. The most common are if, else, and elseif.

Example:

```
<?php
$num = 10;

if ($num > 0) {
  echo "The number is positive.";
} elseif ($num < 0) {
  echo "The number is negative.";
} else {
  echo "The number is zero.";
}
```

```
?>
```

2.4 Loops in PHP

Loops are used to execute the same block of code repeatedly as long as a specified condition is met. Common loops include for, while, do...while, and foreach.

Example of a for loop:

```
<?php
 for ($i = 0; $i < 5; $i++) {
   echo "Value of i: $i<br>";
 }
?>
```

2.5 Functions

Functions are blocks of code that can be repeatedly called and executed whenever needed. They are defined using the function keyword.

Example:

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

 echo greet("Alice"); // Outputs: Hello, Alice!
?>
```

Conclusion.

In the first two chapters of our PHP course, we've laid the foundation for your journey into PHP development. We've introduced PHP, exploring its purpose, history, and how to set up a local

development environment. By understanding the basics, including PHP syntax, variables, data types, operators, and expressions, you've gained essential knowledge needed to write simple PHP scripts.

Moreover, we've delved into control structures such as conditional statements and loops, empowering you to create dynamic and interactive programs. The introduction to functions has provided you with a powerful tool to organize and reuse your code efficiently.

Armed with these fundamentals, you're now ready to tackle more advanced topics and build more complex applications. Keep practicing and experimenting with different PHP concepts to strengthen your understanding. As we progress through the course, you'll continue to enhance your skills and confidence in PHP development.