PHP echo and print

With PHP, there are two basic ways to output data to the screen: echo and print. The differences are small:

- echo has no return value, while print has a return value of 1 so it can be used in expressions
- echo can take multiple parameters, while print can take only one argument
- echo is marginally faster than print

The PHP echo Function

The echo function can be used with or without parentheses: echo or echo(). **Example** echo "Hello"; echo("Hello"); **Example** echo "<h2>PHP is Fun!</h2>"; echo "Hello world!<br>"; echo "I'm about to learn PHP!<br>"; echo "This ", "string ", "was ", "made ", "with multiple parameters.";

Single or Double Quotes?

Strings are surrounded by quotes, but there is a difference between single and double quotes in PHP.

When using double quotes, variables can be inserted to the string as in the example above.

When using single quotes, variables have to be inserted using the . operator, like this:

Example

$txt1 = "Learn PHP";

$txt2 = "PHP";

echo '<h2>' . $txt1 . '</h2>'; echo

'<p>Study PHP at ' . $txt2 . '</p>';

The PHP print Function

The print function can be used with or without parentheses: print or print().

Example

print "Hello"; print("Hello");

Example print "<h2>PHP is

Fun!</h2>"; print "Hello

world!<br>"; print "I'm about to

learn PHP!";

Display Variables

The following example shows how to output text and variables with the print statement:

Example

$txt1 = "Learn PHP";

$txt2 = "php";

print "<h2>$txt1</h2>";

print "<p>Study PHP at $txt2</p>";

Single or Double Quotes?

Strings are surrounded by quotes, but there is a difference between single and double quotes in PHP.

When using double quotes, variables can be inserted to the string as in the example above.

When using single quotes, variables have to be inserted using the . operator, like this:

Example

```
$txt1 = "Learn PHP";

$txt2 = "php";


print '<h2>' . $txt1 . '</h2>'; print

'<p>Study PHP at ' . $txt2 . '</p>';
```

PHP Data Types

PHP Data Types

Variables can store data of different types, and different data types can do different things.

PHP supports the following data types:

- **string** (text values)
- **int** (whole numbers)
- **float** (decimal numbers)
- **bool** (true or false)
- **array** (multiple values)
- **object** (stores data as objects)
- **null** (empty variable)
- **resource** (references external resources)

Use var_dump() to Get the Data Type

To get the data type and the value of a variable, use the var_dump() function.

Example

The var_dump() function dumps the data type and the value:

$x = 5; var_dump($x); //

dumps int(5)

PHP String Data Type

A string is a sequence of characters, like "Hello world!".

Example

$x = 'Hello world!'; var_dump($x);

PHP Int Data Type

An integer data type is a non-decimal number between -2,147,483,648 and 2,147,483,647.

Rules for integers:

- An integer must have at least one digit
- An integer must not have a decimal point
- An integer can be either positive or negative
- Integers can be specified in: decimal (base 10), hexadecimal (base 16), octal (base 8), or binary (base 2) notation

Example $x =

5985;

var_dump($x);

PHP Float Data Type

A float (floating point number) is a number with a decimal point or a number in exponential form.

Example

```
$x = 10.365; var_dump($x);
```

PHP Bool Data Type

A Boolean data type represents two possible states: TRUE or FALSE.

Example

```
$x = true; var_dump($x);
```

Booleans are often used in conditional testing.

PHP Array Data Type

An array data type stores multiple values in one single variable.

In the following example $cars is an array:

Example

```
$cars = array("Volvo","BMW","Toyota");

var_dump($cars);
```

You will learn more about arrays later in this tutorial.

PHP Object Data Type

An object data type holds an instance of a programmer-defined class.

Example

Check the data type of $myCar:

```
class Car {   public $color;   public $model;

public function __construct($color, $model) {
   $this->color = $color;
   $this->model = $model;
  }
  public function message() {     return "My car is a " . $this-
>color . " " . $this->model . "!";
  }
}
$myCar = new Car("red", "Volvo"); var_dump($myCar);
```

PHP NULL Data Type

Null is a special data type which can have only one value: NULL.

A variable of data type NULL is a variable that has no value assigned to it.

Variables can also be emptied by setting the value to NULL:

Example

```
$x = "Hello world!";
$x = null; var_dump($x);
```

Changing Data Type

If you assign an integer value to a variable, the data type will automatically be an integer.

If you assign a string to the same variable, the data type will change to a string:

Example

```
$x = 5; var_dump($x);


$x = "Hello"; var_dump($x);
```

If you want to change the data type of an existing variable, but not by changing the value, you can use casting.

Casting allows you to change data type on variables:

Example

```
$x = 5;

$x = (string) $x; var_dump($x);
```