Pass by Reference

This lesson introduces the ways by which we pass values to a function by reference.

We'll cover the following

What is Pass by Reference?

When passing *arguments* by **pass by reference**, the *original* value is passed. Therefore, the *original* value gets altered. In **pass by reference**, we actually pass the value using *ampersand* sign `&`. The additional `&` is very important: it tells the *compiler* that the data is a *reference* to the *value* rather than simply the value itself.

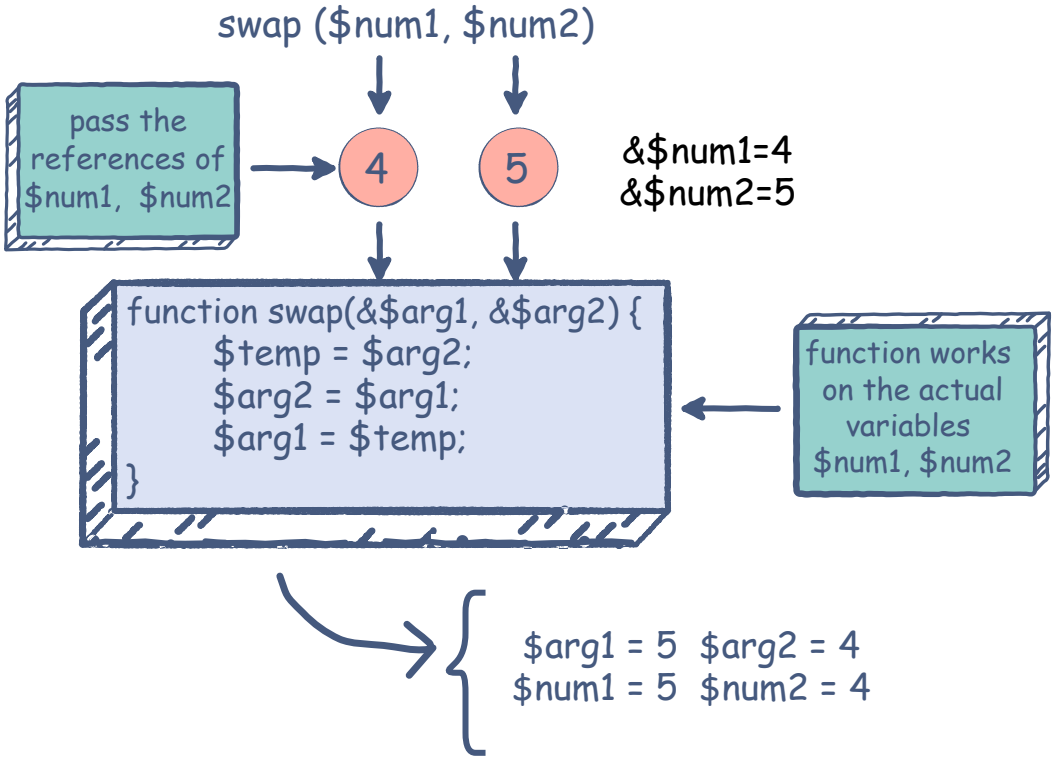
Example: Swap Numbers

Now let's redefine the `interchange` *function* from the example in the previous lesson.

```
<?php
function swap(&$arg1, &$arg2)
{ //parameters num1 and num2 passed using pass by reference method
    $temp = $arg2; //creating a variable temp and setting equal to arg2
    $arg2 = $arg1; //setting the value of arg2 equal to arg1
    $arg1 = $temp; //setting the value of arg1 equal to temp which is equal to arg2

}

$num1 = 4;
$num2 = 5;

// Calling the function with arguments num1 and num2
swap($num1, $num2);
echo "num1 is: $num1\n";
echo "num2 is: $num2";
?>
```

Explanation

The figure below illustrates how this function works:



How values change when passing parameters by reference

- The way the data is passed into `swap` is different from that in our previous example.
- Previously the *arguments* were passed simply as `$num1` and `$num2`, but now they are passed as `&$num1` and `&$num2`.
- The additional `&` is very important: it tells the *compiler* that the data is a **reference** to a value rather than simply that value.

By making use of a *reference* type the *copy* of variables is no longer made.

As an analogy, think of asking someone to proof-read and markup a printed document. You can either give them a photocopy (*pass by value*) or hand them the original document (*pass by reference*). In the same way, you can tell the *compiler* whether to pass a **copy** of the *arguments* or the **original** variables themselves depending on whether you want the originals to be changed or not.

In the upcoming lesson we will discuss the scope of variables in functions.