Programming Basics

Vivify Academy

Functions

What is a function?

A function is a piece of code, a group of instructions, also known as a named procedure, used by programming languages to group some functionality that needs to be used multiple times.

A function takes one or more input values (parameters), does some processing and then returns a single value.

PHP functions

Here are some advantages of using functions:

- Functions reduce the repetition of code within a program,
- Functions make the code much easier to maintain,
- Functions make it easier to eliminate the errors,
- Functions can be reused in other applications

Creating a PHP function

```php
<?php

// Defining the function
function whatIsToday()
{
    echo "Today is " . date('l');
}

// Calling function
whatIsToday();

?>
```

The declaration of a user-defined function start with the keyword **function**, followed by the *name* of the function you want to create, followed by parentheses () and finally place your function's code between curly brackets {}.

A function name must start with a letter or underscore character, not with a number, optionally followed by the more letters, numbers, or underscore characters. Function names are case-insensitive.

PHP functions with parameters

```php
<?php

// Defining a function
function getSum($num1, $num2)
{
    $sum = $num1 + $num2;
    echo "Sum of numbers $num1 and $num2 is : $sum";
}

// Calling a function
getSum(10, 20);

?>
```

You can specify parameters when you define your function to accept input values at run time. The parameters work like placeholder variables within a function; when calling the function, they're replaced by the values (known as arguments) provided to the function.

You can define as many parameters as you like. However for each parameter you specify, a corresponding argument needs to be passed to the function when it is called.

Function parameters vs arguments

An argument is a value that you pass to a function, and a parameter is the variable within the function that receives the argument. However, in common usage these terms are interchangeable i.e. an argument is a parameter is an argument.

Functions with optional parameters and default values

```php
<?php

function eurosToDinars($amount, $exchangeRate = 120)
{
    echo "$amount euros is " .
        $amount * $exchangeRate . " dinars.<br>";
}

eurosToDinars(100);
eurosToDinars(100, 122);

?>
```

You can also create functions with optional parameters — just insert the parameter name, followed by an equals (=) sign, followed by a default value, like this. If you call the function without providing a value for the optional parameter, it's the same as if you provided the default value.

PHP functions returning a value

```
<?php

// Defining a function
function getSum($num1, $num2)
{
    $total = $num1 + $num2;
    return $total;
}

// Printing the returned value
echo getSum(5, 10); // Outputs: 15

$sum = getSum(15, 10);
var_dump($sum);

?>
```

A function can return a value using the return statement followed by a value. Keyword **return** stops the execution of the function ***immediately*** and sends the value back to where the function was called.

This example takes two integer parameters and add them together and then returns their sum to the calling program. Note that return keyword is used to return a value from a function. After the function ends, the function call is “replaced” with the value that the function returned (in this example, **`getSum(15, 10)`** is replaced with the value **`25`**.

PHP functions returning multiple values by returning an array

```
<?php

// Defining function
function divideWithRemainder($a, $b)
{
    $result = $a / $b;
    $remainder = $a % $b;
    return [$result, $remainder];
}

$result = divideWithRemainder(10, 2);
var_dump($result);

?>
```

A function can not return multiple values. However, you can obtain similar results by returning an array, as demonstrated in the following example.

Passing arguments by reference

```
<?php

function addFive($num)
{
    $num += 5;
}

function addSix(&$num)
{
    $num += 6;
}

$originalNumber = 10;

addFive($originalNumber);
echo "Original Value is $originalNumber <br>";

addSix($originalNumber);
echo "Original Value is $originalNumber <br>";

?>
```

In PHP there are two ways you can pass arguments to a function: **by value** and **by reference**. By default, function arguments are passed by value so that if the value of the argument within the function is changed, it does not get affected outside of the function. However, to allow a function to modify its arguments, they must be passed by reference.

Passing an argument by reference is done by prepending an ampersand (&) to the argument name in the function definition, as shown in the example.

Useful links

- http://php.net/manual/en/functions.user-defined.php
- http://php.net/manual/en/functions.arguments.php
- http://php.net/manual/en/functions.returning-values.php
- http://php.net/manual/en/language.references.pass.php
- http://php.net/manual/en/about.prototypes.php

Exercise!