Week 2
Constants

# PHP Constants

If you look up the word constant in a dictionary it will probably tell you that the word is used to describe something that is non-changing and non-variable, and this is exactly the purpose of constants in PHP. A PHP constant is the opposite of a variable in that once it has been defined it cannot be changed.

Constants are particularly useful for defining a value that you frequently need to refer to that does not ever change. For example, you might define a constant called INCHES_PER_YARD that contains the number of inches in a yard. Since this is a value that typically doesn't from one day to the next it makes sense to define it as a constant. Conversely, a value that is likely to change, such as the Dollar to Yen exchange rate is best defined as a variable.

PHP constants are said to have *global scope*. This basically means that once you have defined a constant it is accessible from any function or object in your script.

In addition, PHP provides a number of built-in constants that are available for use to make life easier for the PHP developer.

## Defining a PHP Constant

Rather than using the assignment operator as we do when defining variables, constants are created using the *define()* function. Constants also lack the *$* prefix of variable names.

The define function takes two arguments, the first being the name you wish to assign to the constant, and the second the value to assign to that name.

Constant names are case sensitive. Although it is not a requirement, convention carried over from other programming languages suggests that constants should be named in all upper case characters. The following example demonstrates the use of the *define()* function to specify a constant:

```
<?php
define('INCHES_PER_YARD', 36);
?>
```

Once defined the constant value can be accessed at any time just by referring to the name. For example, if we define a string constant as follows:

```
<?php
define('WELCOME_MESSAGE', "Welcome to my World");
?>
```

We can subsequently access that value anywhere in our script. For example, to display the value:

```
echo WELCOME_MESSAGE;
```

## Checking if a PHP Constant is Defined

It can often be useful to find out if a constant is actually defined. This can be achieved using the PHP *defined()* function. The *defined()* function takes the name of the constant to be checked as an argument and returns a value of *true* or *false* (i.e 1 or 0) to indicate whether that constant exists.

For example, let's assume we wish to find out if a constant named *MY_CONSTANT* is defined. We can simply call the *defined()* function passing through the name, and then test the result using an *if .. else* statement (we will learn if…else statement later on in the course).

```
<?php
define ('MY_CONSTANT',   36);

if (defined('MY_CONSTANT'))
{
     echo "Constant is defined";
}
else
{
     echo "Constant is not defined";
}
?>
```

## Using a Variable as a Constant Name

It is not always the case that you want to *hard-code* a constant name into a script at the point you wish to access it. For example, you may have a general purpose script that you wish to perform tasks on any number of different constants, not just one that happen to have typed in the name for. The best way to resolve this issue is store the name of the constant in a variable. How then, would you access the value assigned to that constant? The answer is to use the PHP *constant()* function. The *constant()* function takes the name of the constant as an argument and returns the value of the constant which matches that name.

The key point to understand here is that the argument passed through to *constant()* can be a string variable set to the name of the constant.

As always an example helps a great deal in understanding a concept. In the script below we define a constant called *MY_CONSTANT*. Next, we create a string variable called constantName and assign it a value of *MY_CONSTANT* (i.e a string that matches the constant name). We can then use this new variable as the argument to *constant()* to obtain the constant value of *MY_CONSTANT*:

<?phpdefine ('MY_CONSTANT', "This is a constant string.");$constantName = 'MY_CONSTANT';if

(defined ( $constantName )){ echo constant($constantName);}else{ echo "$constantName constant is not defined.";}?>

The above script will display the constant value if it exists by using the value of the $constantName variable constant name key.

---

Prev | Table Of Contents | Next