PHP Constants*

Saša Stamenković

Abstract

In this paper the use of constants in php is described. Explanations in the text are followed by examples wich makes a good foundation for further studying about constants. After learning about theoretical concepts pointed out in this paper, one will be able to form and apply his knowledge about constants in practical environments.

In physics "a constant represents a number which expreses quantity or relation that remains unchanged under the specific set of conditions". In mathemathics, a constant is "quantity assumed to remain unchanged within given disccusion". Obvious similarities of the definitions give the introduced term of constant the meaning of something that lasts and remains unchanged in different sets of terms and environments. However, regardless to these similarities in different scientific disciplines, the main subject of these paper are the constants in php and their use in the grouping of certain parameters wich are internal for the script. The process of defining the constants is supported by the unique mechanism that enables forming and usage of the constants.

The mechanisam includes two determing arguments. The first one represents the name of the constant. Naming the constant makes it a special entity. The entity must be clearly structured and characterized by the value given to the name of the constant. The two arguments condition one another.

Finally, a constant represents a name to which is given a scalar value (see table 1). To define a constant we must determine the perameters in the function "define( )".

type of data	**value**
`String`	Alfanumerical value; it may contain any ASCII character number
`Integer`	numerical value; it may be positive, negative or a whole number
`Double (or float)`	floating point value, it may be decimal number
`Boolean`	logical value that can be either true (1) or false (0)

Table 1

Example 1 interprets the scalar values within the above mentioned function:
Example 1

*Version 1.1: Feb 18, 2011 8:53 pm US/Central
†http://creativecommons.org/licenses/by/3.0/

```
<?php
 define ("CONSTANT_NAME", the_value_of_the_constat);
 define ("CONSTANT_NAME2", 10);
 define ("CONSTANT_NAME3", 10.4);
 define ("CONSTANT_NAME4", true);
```

In this way the constant is determined. It is possible to echo out the value in the way described in the following part of the paper:

```
echo CONSTANT_NAME. '</br>';
echo CONSTANT_NAME2. '</br>';
echo CONSTANT_NAME3. '</br>';
echo CONSTANT_NAME4;
?>
```

The result of executing the script is: the_value_of_the_constant.

The purpose of the example 1 is to present the way of forming constants and, also, to point to the possible types of data which can be used in that case. Example 2, described in the following part of the papaer is much more concrete form of the use of the constants.

```
<?php
define ('PI', 3.14159265);
echo PI;
?>
```

Therefore, the result is (3.14159265).

One should be careful about the way of defining the constants which must include a semicolon, This is the right way of doing it, and leaving out a semicolon is not.

```
define (PI, 3.14159265);
```

In defining the constants the usage of capitals is recommendable because it enables making clear difference between the constans and the variables. This can be very useful. Applying this principle will make the constant easier to spot and that will facilitate the job of a programmer.

Finally, there is a limitation that should be pointed to. When naming a constant, a programmer does not have the complete freedom of using certain words such as "Built-In-Keywords". Therefore, if we, for instance, use a keyword "PUBLIC" it is quite certain that we will get the message of syntax error. The same goes for every other reserved word. A complete list of such words can be found at: http://php.net/manual/en/reserved.keywords.php.

In the following part of the paper an exaple will be presented made to generate the constant of the moment of the creation of this particular learning object so that, after executing the script, the date of the creation will be gotten with the aditional generating of the results about the version of php server and the sorts of the operating system that has been used for executing the script.

```
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
1.0 Transitional//EN" "http://www.w3.org/
TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8"
/>
<title>Konstante</title>
</head>
<body>
```

```
<?php
define ('MOMENT', 'february, 19, 2011');
echo '<p>The script has been created on: ' . MOMENT . '.<br />
PHP server version:
<b>'. PHP_VERSION . '</b>  that running on <b>' . PHP_OS . '</b> operating system.</p>';
?>
</body>
</html>
```

In table 2 important structual elements of the presented examples are defined.

Table 2

Prototype	**Construct/function**	**Purpose**
`echo()`	Language construct	shows one or more strings
`define()`	function	defines named constant
`PHP_VERSION`	function	Returns a string containing the version of the currently running PHP parser or extension

Table 2

Literature:

[1] - PHP Cookbook, David Sklar, Adam Trachtenberg, 2nd Edition, O'Reilly, 2006.

[2] - Ajax, JavaScript and PHP All in One, Phil Ballard Michael Moncur, Sams Publishing, 2009.

[3] - PHP 6 and MYSQL 5 for dynamic Web sites, Larry Ullman, Peachpit Press, 2008.

[4] - http://www.php.net