Konstante u PHP-u*

Saša Stamenković

Abstract

U ovom radu definisana je primena konstanti u PHP-u pri radu sa skalarnim vrednostima. Objašnjenja su propraćena konkretnim primerima što omogućava dobru polaznu osnovu za dalji rad sa konstantama. Nakon upoznavanja teorijskih koncepata istaknutih u ovom radu pojedinac će biti osposobljen da formira i primenjuje konstante u praktičnom okruženju.

U fizici „konstanta predstavlja broj koji izražava vlasništvo, kvantitet ili relaciju koja ostaje nepromenjena pod specifičnim uslovima“. U matematici - reč je o „kvantitetu za koji se pretpostavlja da će ostati nepromenjen unutar postavljene diskusije“. Konstante u PHP-u su „imena čije se vednosti ne mogu menjati u procesu izvršenja skripte“. Očigledne sličnosti izloženih definicija povezuju uvedeni pojam konstanti kao nešto što ima cilj da traje, opstaje i ostaje nepromenjeno u različitim uslovima ili sredinama. No, bez obzira na sličnosti tematskog područja, težište ovog rada je ipak usmereno na konstante u PHP-u, i u tom kontekstu, opravdanje za njihovu primenu se može naći u procesu grupisanja pojedinih paramatara koji su interni za skript. Proces definisanja konstanti je podržan jedinstvenim mehanizmom pomoću koga one nastaju i bivaju upotrebljene.

Pomenuti mehanizam uključuje dva determinišuća argumenta: prvi reprezentuje ime konstante čijim se dodeljivanjem ona deklariše. Na ovaj način konstanta postaje zasebni entitet. Entitet mora imati jasnu stukturu ili karakter. U ovom slučaju radi se o vrednosti koja se dodeljuije imenu. Ovde sledi da postoji međusobna uslovljenost prvog i drugog argumenta.

Konačno, konstanta predstavlja ime kojem je pridružena skalarna vrednost (videti tabelu 1). Da bismo definisali konstantu, moramo najpre odrediti parametre u funkciji define();.

Tipovi podataka

tip podatka	vrednost
`String`	Alfanumerička vrednost; može sadržati bilo koji broj ASCII karaktera
`Integer`	numerička vrednost; može biti pozitivna, negativna ili ceo broj

continued on next page

*Version 1.2: Feb 18, 2011 8:29 pm US/Central
†http://creativecommons.org/licenses/by/3.0/

`Double (ili float)`	vrednost sa pokretnim zarezom, može biti decimalni broj
`Boolean`	logička vrednost koja može biti ili tačna (1) ili netačna (0)

Table 1

Primer 1 interpretira skalarne vrednosti unutar pomenute f-je:

Primer 1

```
<?php
 define ("IME_KONSTANTE", vrednost_konstante);
 define ("IME_KONSTANTE2", 10);
 define ("IME_KONSTANTE3", 10.4);
 define ("IME_KONSTANTE4", true);
```

Na ovaj način konstanta je deklarisana. Pozivanje vrednosti konstante moguće je izvršiti na način opisan u nastavku:

```
echo IME_KONSTANTE. '</br>';
echo IME_KONSTANTE2. '</br>';
echo IME_KONSTANTE3. '</br>';
echo IME_KONSTANTE4;
?>
```

Izvršavanje skripte daje za rezultat: vrednost_konstante. Primer 1 ima za cilj da pretstavi način formiranja konstani, ali i da ukaže na moguće tipove podataka koji se pri tom mogu koristiti. Primer 2, dat u nastavku, znatno je konkretniji oblik primene konstanti.

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

Rezultat, je dakle, (3.14159265).

Treba voditi računa o načinu definisanja konstanti koja respektivno uključuje jednostruke znake navoda ('). Ovaj način je ispravan, dok izostavljanje navodnika nije!

define (PI, 3.14159265);

Poželjno je da prilikom definisanja konstanti budu korišćena velika slova. Na ovaj način se jasno mogu razlikovati konstante od promenljivih, što ume da bude od velike koristi u praksi. Primenjujući ovaj princip one će biti znatno uočljivije, što će u brojnim situacijama olakšati rad programeru.

U uvodnim razmatranjima ukratko je napomenuto da su konstante obično imena kojima su pridružene skalarne vrednosti, da bi nakon toga bile imenovane pojedine konstante u kontekstu praktičnih primera. Na kraju su istaknuti i principi dobre prakse kojih se treba pridržavati.

Konačno, treba ukazati na jedno važno ograničenje! Prilikom imenovanja konstanti programer nema potpunu slobodu budući da ne sme koristiti pojedine rezervisane reči koje se u svom izvornom obliku nazivaju „Built-in Keywords“, pa tako, ukoliko upotrebimo Keyword PUBLIC, sasvim je izvesno da ćemo dobiti sintaksnu grešku. Isto važi i za sve druge rezervisane reči, a moguće je upoznati se sa njihovom punom listom na: http://php.net/manual/en/reserved.keywords.php

U nastavku je dat konačan primer koji je projektovan tako da generiše konstantu trenutka nastanka ovog learning object-a tako da se nakon izvršavanja skripte dobija datum kreiranja uz dodatno generisanje rezultata o verziji php servera i vrsti operativnog sistema na kojoj se skripta izvršava

```
    <!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 ('TRENUTAK', '10. februara, 2011. god.');
echo '<p>Skripta je kreirana: ' . TRENUTAK . '.<br />
Verzija PHP servera:  <b>'. PHP_VERSION . '</b>  koji radi
na <b>' . PHP_OS . '</b> operativnom sistemu.</p>';
?>
</body>
</html>
```

U tabeli 2 definisani su važniji gradivni elementi prezentovanih primera.

Prototip	**Konstrukt/funkcija**	**svrha**
`echo()`	Jezički konstrukt	prikazuje jedan ili više stringova
`define()`	funkcija	definiše imenovanu konstantu
`PHP_VERSION`	funkcija	Vraća string koji prikazuje trenutnu verziju PHP parsera ili ekstenzije

Table 2

Slede: "Izrazi, operatori i dodeljivanje vrednosti promenljivama"

Literatura

[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