Tipovi podataka, upotreba izraza i operatora

Tipovi podataka

- Različiti tipovi podataka zauzimaju različite memorije i mogu da budu tretirani drugačije kada skript njima manipuliše.
- PHP je jezik sa **nesigurno definisanim tipovima**, što znači da automatski određuje tip podataka u vreme kada je podatak dodeljen svakoj promenljivoj.
- Automatska dodela tipa podataka je ponekad dobra, a ponekad nije.
- S jedne strane, to znači da promenljive mogu da budu upotrebljene fleksibilno – u jednom primeru promenljiva može da sadrži znakovni niz, a kasnije u skriptu ceo broj ili neki drugi tip podatka.
- S druge strane, fleksibilnost može da dovede do problema u većim skriptovima ako konkretno očekujete da promenljiva sadrži jedan tip podatka, a ona u stvari sadrži nešto sasvim drugo.

Tipovi podataka

Tip	Primer	opis
Logički tip	True	Jedna od specijalnih vrednosti true ili false
Ceo broj	5	Ceo broj
Brojevi u pokretnom zarezu	3.234	Broj sa pokrenim zarezom
Znakovni niz	“programiranje”	Kolekcija znakova
Objekat		Instanca klase
Niz		Uređeni skup ključeva i vrednosti
Resurs		Referenca ka nezavisnom resursu (npr. bazi podataka)
Null		Nepokrenuta promenljiva

Tipovi podataka
is_*

- PHP ima nekoliko dostupnih funkcija za testiranje validnosti određenog tipa promenljive – po jednu za svaki tip.
- Familija funkcija is_* testira da li je data vrednost određenog tipa podataka.
- Na primer funkcija is_bool() testira da li je data vrednost logička.

```php
<?php

$a=true;

print "da li je promenljiva $a boolean? ". is_bool($a);



//Program izbacuje:
// da li je promenljiva 1 boolean? 1
```

Rezultat 1 znači **true**.

Tipovi podataka
is_*

Familija funkcija **is_***:

- **is_null()** – ispituje da li je promenljiva null
- **is_int()** – ispituje da li je promenljiva ceo broj
- **is_string()** – ispituje da li je promenljiva znakovni niz
- **is_double()** – ispituje da li je broj u pokretnom zarezu
- **is_bool()** – ispituje da li je promenljiva logički tip
- **is_array()** – ispituje da li je promenljiva niz
- **is_numeric()** – ispituje da li je promenljiva broj ili numerički niz
- **is_resource()** – ispituje da li je promenljiva resurs

Tipovi podataka
is_*

```php
<?php

$a; //$a ima NULL vrednost

print "da li je NULL? " . is_null($a);
print "<BR>";
$a=1;
print "da li je integer? " . is_integer($a);
print "<BR>";
$a=1.1;
print "da li je double? " . is_double($a);
print "<BR>";
$a="Baze podataka";
print "da li je string? " . is_string($a);
?>
```

Program izbacuje:

da li je NULL? 1
da li je integer? 1
da li je double? 1
da li je string? 1

Tipovi podataka
settype()

- PHP obezbeđuje i funkciju **settype()**, koja se koristi za menjanje tipa promenljive.

Funkcija izgleda: **`settype($a, 'novi tip')`**

- **$a** je promenljiva čiji tip želimo da promenimo.
- **'novi tip'** je tip u koji želimo da promenimo promenljivu **$a**.

```php
<?php

$a=3.14;
print "da li je promenljiva $a decimalni broj? " .is_double($a);
print "<BR>";
settype( &var: $a, type: 'integer');
print "da li je promenljiva $a ceo broj? " .is_integer($a);
print "<BR>";
?>
```

Program izbacuje:

da li je promenljiva 3.14 decimalni broj? 1
da lie je promenljiva 3 ceo broj? 1

Tipovi podataka
eksplicitne konverzije

Eksplicitna konverzija isto menja tip podatka promenljive kao i funkcija **settype()**. Razlika je u tome što eksplicitnom konverzijom **kreiramo kopiju**, ostavljajući originalnu promenljivu netaknutom.

Eksplicitna konverzija izgleda ovako:

`$a = (tip) $b`, npr: **`$a = (integer) $b`**

Ako je **$b = 3.14**; onda je **$a = 3**; promenljiva **$b** je ostala nepromenjena.

```php
<?php

$b=3.14;
$a=(integer)$b;

print "a = ".$a;
print "<BR>";
print "b = ".$b;
?>
```

Program izbacuje:

a = 3
b = 3.14

Tipovi podataka
eksplicitne konverzije

Tipovi konverzije:

- (double)
- (string)
- (integer)
- (boolean)
- Funkcija **gettype($a)** vraća tip promenljive **$a**

```php
<?php

$a= 3.14;
print "tip promenljive a je: ".gettype($a);
?>
```

Program izbacuje:

tip promenljive a je: double

Tipovi podataka
eksplicitne konverzije

U toku eksplicitne konverzije znakovnog niza u ceo broj ili decimalni, PHP ignoriše sve nenumeričke karaktere.

Znakovni niz je skraćen i svi karakteri od lokacije prvog nenumeričkog karaktera se ignorišu.

```php
<?php

$a="30cm";

$b=(int)$a;

print $b; //vidimo da prilikom eksplicitne konverzije cm se gubi

print "<BR>";

$a= "1m 30cm";

$b=(int) $a;

print $b; //vidimo da prilikom ekplicitne konverzije se sve vrednosti gube posle prvog nenumerickog karaktera

?>
```

Program izbacuje:

30
1

Upotreba izraza i operatora

Operatori u PHP-u su simboli koji se koriste za manipulisanje podacima koji se čuvaju u promenljivim, za omogućavanje upotrebe jedne ili više vrednosti za kreiranje nove vrednosti, za proveru validnosti podataka za određivanje sledećeg koraka u uslovu, itd.

Vrednosti na koje deluje operator naziva se **operand.**

Izraz: **4+5**

- 4 i 5 su **operandi**, a + je **operator**
- Kombinacija operanda sa operatorom za kreiranje rezultata naziva se **izraz**.

Upotreba izraza i operatora
operator dodele (=)

Operator dodele (=) koristi vrednost operanda sa desne strane i dodeljuje je operandu sa leve strane.

Primer: **`$name= "Marija";`**

Promenljiva $name sada sadrži znakovni niz Marija.

Upotreba izraza i operatora
aritmetički operatori

Aritmetički operatori izvršavaju aritmetičke operacije:

Operator	Naziv	Primer
+	Sabiranje	10+5
-	Oduzimanje	10-5
/	Deljenje	10/5
*	Množenje	10*3
%	Modulus	10%3

Upotreba izraza i operatora
operator nadovezivanja

Operator nadovezivanja je predstavljen u PHP-u jednom tačkom (.).

Kada se tretiraju oba operanda kao znakovni nizovi, ovaj operator dodaje operand sa desne strane operandu sa leve strane.

```php
<?php

$a="Baze";
$b="podataka";

print "$a "."$b"; //namerno razmak posle promenljive $a da reci ne bi bile spojene

?>
```

Program izbacuje: Baze podataka

Upotreba izraza i operatora
operator nadovezivanja

Bez obzira na tipove podataka operanda koji su upotrebljeni sa operatorom nadovezivanja, oni se tretiraju kao znakovni nizovi, a rezultat je uvek tip znakovnog niza.

Upotreba izraza i operatora
operator nadovezivanja

```php
<?php

$a= 212;

print "Visina je ".($a/100)." metara"

?>
```

Program izbacuje:

Visina je 2.12 metara

Upotreba izraza i operatora
kombinovani operatori dodele

Kombinovani operator dodele se sastoji od standardnog simbola operatora, iza kojeg se nalazi znak jednakosti.

Kombinacija operatora dodele uštedeće upotrebu dva operatora u dva različita koraka unutar skripta.

```php
<?php
// imamo promenljivu koja ima vrednost 4
// i zelimo da je povecamo za jos 4

$a=4;
$a=$a+4;
print $a;
print "<BR>";
print "ili";
print "<BR>";
$b=4;
$b+=4;
print $b;
?>
```

Program izbacuje:

8
ili
8

Upotreba izraza i operatora
kombinovani operatori dodele

Operator	Primer	Ekvivalent za
+=	$x+=5	$x=$x+5
-=	$x-=5	$x=$x-5
/=	$x/=5	$x=$x/5
*=	$x*=5	$x=$x*5
%=	$x%=5	$x=$x%5
.=	$x.=" test"	$x=$x." test"

Upotreba izraza i operatora
automatsko povećanje i smanjivanje promenljive celog broja

Operator za **sufiksno inkrementiranje ++**, povećava vrednost promenljive za 1.

Operator za **sufiksno dekrementiranje --**, smanjuje vrednost promenljive za 1.

```php
<?php
$a=5;
$b=10;
$a++;
$b--;
print $a;
print "<BR>";
print $b;
?>
```

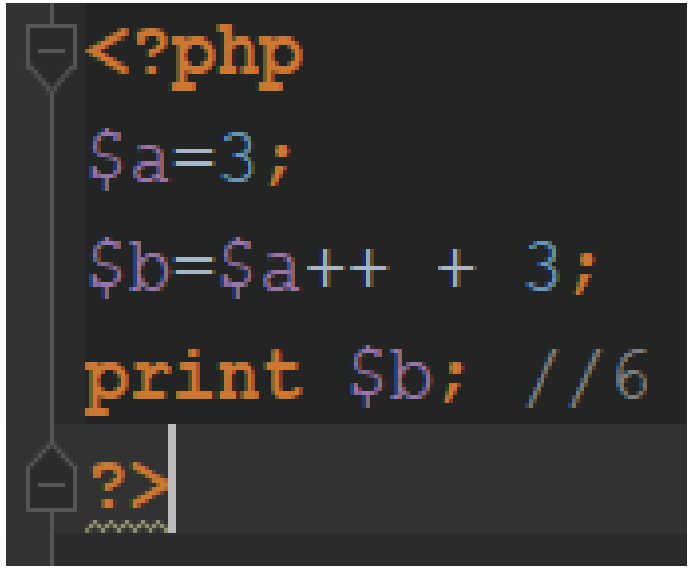
Program izbacuje:

6
9

Upotreba izraza i operatora
automatsko povećanje i smanjivanje promenljive celog broja

Ako se upotrebi operator za sufiksno inkrementiranje ili za sufiksno dekrementiranje zajedno sa uslovnim operatorom, operand je modifikovan samo kada je završena prva operacija.

U ovom primeru promenljiva $b prvo postaje 6 (rezultat od 3 + 3), a zatim je promenljiva $a povećana.



Upotreba izraza i operatora
automatsko povećanje i smanjivanje promenljive celog broja

U nekim okolnostima je potrebno povećanje ili smanjenje promenljive u izrazu testa pre nego što se izvrši testiranje.

PHP za to obezbeđuje operatore za **prefiksno inkrementiranje** i za **prefiksno dekrementiranje**.

```php
<?php
$a=3;
$b=++$a + 3;
print $b; //7
?>
```

Upotreba izraza i operatora
operatori poređenja

Operatori poređenja izvršavaju testove poređenja koristeći njihove operande i vraćaju logičku vrednost **true**, ako je test uspešan ili **false** , ako je test neuspešan.
Ovaj izraz je koristan kada se koriste strukture kontrole u skriptovima, kao što su iskazi **if** i **while.**

```php
<?php
 //Koriscenje operatora poredjenja

$a=5;

print $a<7; // posto $a ima vrednost 5 ovaj izraz je true
print "<BR>";
print $a<5; //ovaj izraz je false




?>
```

Program izbacuje:

1

Upotreba izraza i operatora
operatori poređenja

Operator	Naziv	Primer ($x je 4)	rezultat
==	Jednako	$x==5	False
!=	Nejednako	$x!=5	True
===	Identično	$x===4	True
!==	Neekivalentno	$x!==”4”	False
>	Veći od	$x>4	False
>=	Veći ili jednak od	$x>=4	True
<	Manji od	$x<4	False
<=	Manji ili jednak od	$x<=4	True

Upotreba izraza i operatora
logički operatori

Logički operatori testiraju kombinacije logičkih vrednosti.

Operator **or** piše se sa dve uspravne crtice (||).

```php
<?php

$a=5;

print ($a>10 || $a==5)  //Vraca vrednost 1 zato sto je $a=5

?>
```

Program izbacuje:

1

Upotreba izraza i operatora
logički operatori

Operator **and** se označava (&&).

```php
<?php

$a=5;

print ($a>3 && $a<7)  //Vraca vrednost 1

?>
```

Operator	Naziv	Vraća true …	Primer	rezultat
||	or	Ako je levi ili desni tačan	true||false	true
or	or	Ako je levi ili desni tačan	true or false	true
xor	Xor	Ako je levi ili desni tačan, ali ne i oba	true xor true	false
&&	And	Ako su levi i desni tačni	true && false	false
and	And	Ako su levi i desni tačni	true and false	false
!	Not	ako je jedan operand netačan	!true	false

Upotreba izraza i operatora
prioritet operatora

PHP koristi različite prioritete za različite operatore:

Na primer operator * ima prednost u odnosu na operator +. Međutim može se promeniti prioritet operatora postavljanjem zagrada oko izraza: (4+5)*2

Prvo će se izvršiti sabiranje, a zatim oduzimanje.

```php
<?php

print 5+4*2; //Vraca vrednost 13
print "<BR>";
print (5+4)*2; //Vraca vrednost 18

?>
```

Upotreba izraza i operatora
prioritet operatora

Sledi spisak operatora koji su raspoređeni po prioritetu:

1. ++, --, (cast)
2. /, *, %
3. +, -
4. <, <=, =>, >
5. ==, ===, !=
6. &&
7. ||
8. =, +=, -=, /=, *=, %=, .=
9. and
10. xor
11. or

Upotreba izraza i operatora
prioritet operatora

Kao što možete da se vidi, **or** ima niži prioritet od operatora **||**, a **and** od operatora **&&**, pa mogu da se upotrebe logički operatori nižeg prioriteta da bi promenili način čitanja izraza složenog testa.

Napomena: u većini slučajeva upotreba zagrada kod čini jednostavnijim i sa manje grešaka od koda koji koristi razliku u prioritetima ovih operatora.

Konstante

Ako hoćemo da koristimo vrednost koja mora da ostane nepromenjena prilikom izvršenja skripta, može da se definiše i upotrebi **konstanta**.

Za kreiranje konstante, koristi se PHP ugrađena funkcija **define()** za kreiranje konstante koja naknadno ne može da bude promenjena, osim ako se ponovo upotrebi funkcija **define()**.

```php
<?php

define("predmet", "baze podataka");
print predmet; //baze podataka


?>
```

Konstante

Funkcija **define()** takođe prihvata treći logički argument koji određuje da li naziv konstante treba da razlikuje velika i mala slova. Prema standardnom podešavanju, nazivi konstanti razlikuju velika i mala slova. Međutim prosleđivanjem vrednosti **true** u funkciji **define()** može se promeniti ovo ponašanje.

```php
<?php

define("PREDMET", "baze podataka",true);
print predmet; //baze podataka


?>
```