Lekcija 05

OSNOVNI POJMOVI I PARAMETRI U PHP

dr Miroslava Raspopović, Andrej Stanišev, Jovana Kovač

OSNOVNI POJMOVI I PARAMETRI U PHP

Uvod

Uvod

- *Sadržaj*

01

Uvod u PHP

- *Osnovni pojmovi*
- *PHP komentari*
- *PHP promenljive*
- *Aritmetički i logički operatori*
- *Kontrola toka*
- *Petlje*
- *Funkcije*

02

PHP parametri

1. *HTML forme i PHP*
2. *Primeri i rešenja zadataka*

UVOD

Cilj ovog predavanja je da objasni osnovne funkcionalnosti rada PHP jezika, u kobinaciji sa HTML-om i CSS-om koji su bili obrađivani u prethodnim lekcijama

U okviru ove lekce biće reči o sledećim temama:

- Osnovni pojmovi PHP-a
- PHP komentari
- PHP promenljive
- Aritmetički i logički operatori
- Kontrola toka
- Petlje
- Funkcije
- HTML forme i PHP

Uvod u PHP

- *Osnovni pojmovi*
- *PHP komentari*
- *PHP promenljive*
- *Aritmetički i logički operatori*
- *Kontrola toka, ...*

01

OSNOVNI POJMOVI

PHP je skript jezik opšte namene koji je pogodan za Web razvoj

PHP je skript jezik opšte namene koji je pogodan za Web razvoj. Po nekim istraživanjima, PHP se nalazi na četvrtom mestu najpopularnijih programskih jezika, odmah ize Jave, C i Visual Basic jezika. PHP se izvršava na Web serveru gde se kao ulaz pojavljuje PHP kod, a izlaz je Web strana.

Postoje programski paketi koji omogućavaju kreiranje dinamičkih Web strana na računaru koji ne mora da bude povezan na WWW. WAMP rešenja su paketi programa kreiranih za Windows OS. WAMP je akronim, kreiran od naziva operativnog sistema (Windows) i osnovnih komponenti paketa: Apache, MySQL i PHP.

Apache je Web server koji omogućava pregled Web strana uz pomoć standardnih čitača poput Internet Explorer-a ili Firefox-a. MySQL je program za upravljanje bazama podataka.

PHP je jezik koji omogućava manipulaciju podacima u bazi i generisanje Web strana. Slično, postoji i LAMP rešenje za Linux OS. Za razliku od samog Windows OS, komponente WAMP/LAMP paketa su open source.

PRVI PHP PROGRAM

PHP skript počinje sa <?php, a završava se sa ?>.

Blok u dokumentu koji se odnosi na PHP skript počinje sa <?php, a završava se sa ?>. Ovaj blok može se nalaziti bilo gde u HTML dokumentu. PHP datoteka obično sadrži HTML tagove i dodatni skript kod.

Primer:

Kreirati PHP skript koji će u čitaču prikazati poruku: „Pozdrav!“.

Postoje dve komande za slanje i prikazivanje nekog teksta u čitaču: print i echo.

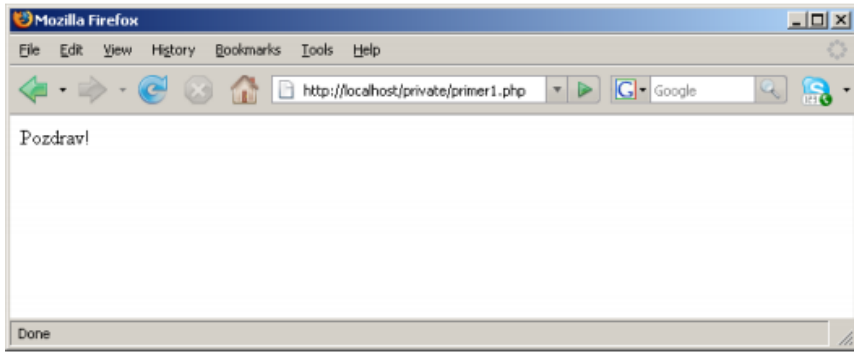
Dokument može da izgleda ovako:

```
<html>
<body>
<?php
echo "Pozdrav!";
?>
</body>
</html>
```

Svaka linija koda završava se sa tačkom-zarez. To je separator i koristi se da razdvoji jedan skup instrukcija od drugih. Sadržaj se na već poznati način unosi u Notepad i da bi mogao biti prikazan uz pomoć phpdev paketa potrebno je da se snimi na adresu: c:\phpdev\private\. Dokument snimamo kao npr. "primer1.php".
Da bi dokument bio pravilno prikazan potrebno je da pokrenemo phpdev na sledeći način:

Start→All programs→phpdev→phpdev_2K_XP_NT.

Nakon toga u čitač se unosi sledeća adresa:
http://localhost/private/primer1.php.
Pojaviće se snimljeni dokument.



Slika-1 Snimljeni dokument

KOMENTARI

U PHP jeziku znaci // i / */ se koriste za komentare*

U PHP jeziku znaci // se koriste za komentare koji se nalaze u jednom redu, a znaci /* i */ za duže komentare.

```
<html>
<body>
<?php

//Ovo je komentar

/*Ovo je
duži
komentar
*/
?>
</body>
</html>
```

PROMENLJIVE

Promenljive se koriste za čuvanje vrednosti, kao što su alfa-numerički znaci, brojevi ili nizovi

Promenljive se koriste za čuvanje vrednosti, kao što su alfa-numerički znaci, brojevi ili nizovi. Kada se uvede, promenljiva, odnosno njena vrednost, se može koristiti u datom skriptu uvek kada je to potrebno. Sve promenljive počinju znakom $. Pravilan način korišćenja promenljive:

$ime_promenljive = vrednost;

Čest je slučaj da se tokom pisanja skripta izostavi znak $. U tom slučaju skript neće funkcionisati. Na primer, ako želimo da u skriptu koristimo jednu string i jednu celobrojnu promenljivu, to može da izgleda ovako:

```
<?php
$txt = "FIT";
$broj = 54;
?>
```

U PHP jeziku nije potrebno deklarisati promenljive pre nego što se koriste. PHP automatski konvertuje promenljivu u odgovarajući tip, u zavisnosti od načina njihovog korišćenja. Naziv promenljive mora na početku da sadrži slovo ili znak underscore (_).

Na ostalim mestima u nazivu mogu se naći mala i velika slova, underscore ili brojevi. Naziv ne bi trebalo da sadrži razmake. Ako se, ipak, sastoji od više od jedne reči one se razdvajaju znakom underscore ili velikim slovima (npr. **$ovo_je_promenljiva** ili **ovoJePromenljiva**).

ARITMETIČKI OPERATORI I OPERATORI DODELJIVANJA

U PHP-u se mogu koristiti aritmetički operatori. Operatori dodeljivanja dodeljuju vrednost promenjivoj

U PHP-u se mogu koristiti različiti operatori, dati na sledećim slikama:

Operator	Opis	Primer	Rezultat
+	Sabiranje	x = 3 x + 2	5
-	Oduzimanje	x = 3 6 - x	3
*	Množenje	x = 3 x * 6	18
/	Deljenje	6 / 3	2
%	Modul	6 % 3 8 % 3	0 2
++	Inkrement	x = 3 x++	x = 4
--	Dekrement	x = 3 x--	x = 2

Slika-2 Aritmerički operatori

Operatori dodeljivanja:

Operator	Primer	Rezultat
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x – y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
.=	x .= y	x = x.y
%=	x %= y	x = x % y

Slika-3 Operatori dodeljivanja

OPERATORI POREĐENJA

Operatori poredjenja porede vrednosti

Operatori poredjenja:

Operator	Opis	Primer
==	jednako	3 == 6 daje rezultat **false**
!=	nije jednako	3 != 6 daje rezultat **true**
>	veće od	3 > 6 daje rezultat **false**
<	manje od	3 < 6 daje rezultat **true**
>=	veće od ili jednako	3 >= 6 daje rezultat **false**
<=	manje od ili jednako	3 <= 6 daje rezultat **true**

Slika-4 Operatori poredjenja

LOGIČKI OPERATORI

Logički operatori izvršavaju logičke operacije nad operandima

Logički operatori:

Operator	Opis	Primer
&&	logičko I	x = 3 y = 6 (x < 9 && y > 0) daje rezultat **true**
||	logičko ILI	x = 3 y = 6 (x == 3 || y == 3) daje rezultat **true**
!	negacija	x = 3 y = 6 !(x == y) daje rezultat **true**

Slika-5 Logički operatori

KONTROLA TOKA (IF ELSE)

If-else konstrukcija se koristi za kontrolu toka programa

Često tok izvršenja nekog programa zavisi od ispunjenja određenih uslova, odnosno izvršavaće se različite instrukcije programa, u zavisnosti od definisanih odluka. Ove odluke se zovu i uslovne naredbe. Naredba if...else se koristi kada je potrebno izvršiti skup nekih instrukcija kada je ispunjen zadati uslov, odnosno neki drugi skup instrukcija ako taj uslov nije ispunjen. Naredba elseif se koristi u kombinaciji sa if...else naredbom kada je potrebno izvršiti deo programa ako je ispunjen jedan od navedenih uslova.

Opšti slučaj if...else naredbe izgleda ovako:

```
if (uslov)
kod koji će se izvršiti ako je uslov ispunjen;
else
kod koji će se izvršiti ako uslov nije ispunjen;
```

PRIMER KONTROLE TOKA

U if i else blokovima se mogu nalaziti jedna ili više linija koda

Primer:

Kreirati PHP skript koji će u čitaču prikazati tekst: „Odlično!“, ako je domaći zadatak dobro urađen. U suprotnom, prikazaće se tekst: „Žao mi je. Zadatak nije dobar.“.

Rešenje:

Dokument može da izgleda ovako:

```
<html>
<body>
<?php
$zadatak="dobar";
if ($zadatak=="dobar")
echo "Odlično!";
else
echo "Žao mi je. Zadatak nije dobar.";
?>
</body>
</html>
```

Ako se kod snimi npr. kao "primer2.php", od vrednosti promenljive **$zadatak**, zavisiće i prikaz u čitaču.

Ako bismo želeli da se u slučaju da zadatak nije dobar, tekst ispiše u dva reda, potrebno je koristiti zagrade { i } na sledeći način:

```
<html>
<body>
<?php
$zadatak="dobar";

if ($zadatak=="dobar")
echo "Odlično!";

else
{
echo "Žao mi je. <br/>“;
echo "Zadatak nije dobar.";
}
?>
</body>
</html>
```

ELSEIF KONSTRUKCIJA

Elseif konstrukcija je kombinacija else i if konstrukcija

U slučajevima kada je potrebno izvršiti deo koda ako je ispunjen jedan od nekoliko uslova koristi se elseif naredba na sledeći način:

```
if (uslov1)
kod koji će se izvršiti ako je uslov1 ispunjen;
elseif (uslov2)
kod koji će se izvršiti ako je uslov2 ispunjen;
else
kod koji će se izvršiti ako uslov nije ispunjen;
```

Sledi primer elseif konstrukcije. U ovom slučaju ako je ispunjen uslov da je zadatak dobar i da se radi o 15. domaćem zadatku, pojaviće se prva poruka.
Ako je samo ispunjen uslov da je zadatak tačan, u čitaču će se pojaviti drugi tekst.
Konačno, ako nije ispunjen nijedan od prethodna dva uslova pojaviće se informacija da zadatak nije dobar.

```
<html>
<body>
<?php
$redniBroj=15;
$zadatak="dobar";

if ($redniBroj==15 && $zadatak==“dobar“)
echo "Čestitam! Uradili ste sve domaće zadatke.";
elseif ($zadatak=="dobar")
echo "Odlično!“;

else
{echo "Žao mi je. <br/>“;
echo "Zadatak nije dobar.";
}
?>
</body>
</html>
```

WHILE PETLJA

While petlja je osnovna konstrukcija za ponavljanje jednog dela koda

U programiranju je čest slučaj da je potrebno jedan isti kod izvršavati više puta. PHP za te namene koriste petlje: while, do...while, for i foreach.

Petlja while će se izvršavati sve dok je ispunjen navedeni uslov:

```
while (uslov)
kod koji će se izvršiti;
```

Primer:

Koristeći while petlju ispisati nazive 15. nedelja u semestru u formatu: „x. nedelja“, gde je x redni broj nedelje.

```
<html>
<body>
<?php
$i=1; while($i<=15)
{
echo $i.“. nedelja<br/>";
$i++;
}
?>
</body>
</html>
```

Na početku skripta je definisana promenljiva i, kojoj je dodeljena vrednost 1. Kod koji se nalazi u while petlji, izvršavaće se sve dok je ispunjen uslov da je i ≤ 15, pri čemu se svakim izvršavanjem petlje vrednost promenljive uvećava za jedan ($i++).
Pre poslednjeg izvršavanja petlje promenljiva i se uvećava za jedan i postaje jednaka petnaest. Nakon prikazivanja te vrednosti, vrednost promenljive se ponovo povećava za jedan (i = 16), pa uslov za izvršavanje petlje više nije ispunjen. što znači da skript prestaje sa izvršavanjem.
U delu sa echo komandom, za spajanje prikaza vrednosti promenljive i teksta: „. nedelja“, koristili smo tačku. To je tzv. concatenation operator koji npr. spaja dva teksta u jedan prikaz.

DO-WHILE PETLJA

Do-while petlja se koristi kada je telo petlje potrebno izvršiti bar jednom

Kod while petlje, izvršavanje koda unutar petlje nije obavezno, odnosno ako uslov nije na početku ispunjen kod se neće izvršiti.

To bi se desilo da smo definisali na početku da je npr. $i=16. Petlja do...while, za razliku od while petlje, izvršiće kod bar jednom. U opštem slučaju, do...while petlja izgleda ovako:

```
do
{
kod koji će se izvršiti;
}
while (uslov);
```

Na primer, prethodni primer bi u tom slučaju mogao da izgleda ovako:

```
<html>
<body>
<?php
$i=0;

do
{
$i++;
echo $i.“. nedelja<br/>";
}

while ($i<15);
?>
</body>
</html>
```

FOR PETLJA

For petlja je kraća verzija while petlje

Na sličan način se koristi i **for** petlja. Ova komanda se koristi kada se unapred zna broj izvršavanja. Opšti oblik izgleda ovako:

```
for (inicijalizacija; uslov; inkrement)
{
kod koji će se izvršiti;
}
```

Petlja for ima tri parametra. Prvi parametar inicijalizuje promenljivu, drugi predstavlja uslov, a treći sadrži inkrement potreban za izvršavanje petlje.

Inicijalno se promenljivoj dodaje vrednost jedan, i ona se u toku izvršavanje petlje inkrementalno povećava za jedan ($i++) sve dok je ispunjen uslov da je: $i<=15.
Korišćenje ove petlje za poslednji primer ispisivanja naziva nedelja može da izgleda ovako:

```
<html>
<body>
<?php
for ($i=1; $i<=15; $i++)
{
echo $i.“. nedelja<br/>";
}
?>
</body>
</html>
```

FUNKCIJE

Funkcija se definiše koristeći ključnu reč function

Sintaksa funkcija u PHP jeziku je slična kao u JS.

```
function foo($arg_1, $arg_2, /* ..., */ $arg_n)
{
    echo "Example function.\n";
    return $retval;
}
```

Funkcija se definiše koristeći ključnu reč function. U zagradi se nabrajaju parametri funkcije.
Ključna reč return vraća vrednost iz funkcije.
Prilikom poziva funkcije, prosledjuju se stvarni parametri. Na primer,

```
foo(1,"asdf",5)
```

PHP parametri

1. HTML forme i PHP

2. Primeri i rešenja zadataka

02

HTML FORME I PHP

HTML forme su način da se podaci prikupe od korisnika i proslede prema nekom udaljenom serveru, a zatim dalje obrade

HTML forme su način da se podaci prikupe od korisnika i proslede prema nekom udaljenom serveru, a zatim dalje obrade. Jedan od primera sa kojim se korisnici često sreću jeste korišćenje nekog pretraživača. Na primer, osnovu korisničkog interfejsa Google pretraživača čini Web forma sa jednim poljem za unošenje pojmova na osnovu kojih se vrši pretraživanje i dva dugmeta za prosleđivanje.

Jedna od stvari na koju posebno treba obratiti pažnju prilikom rada sa PHP jezikom je mogućnost automatskog korišćenja sadržaja neke HTML forme u PHP skriptu.

PRIMER 1

Kreiranje HTML forme za automatski prikaz imena i starosti korisnika

Zadatak:

Kreirati HTML formu i odgovarajući PHP skript koji će omogućiti da se automatski prikažu ime i starost korisnika. Formu čine dva polja, jedno za unos imena i drugo za unos godina.

Rešenje:

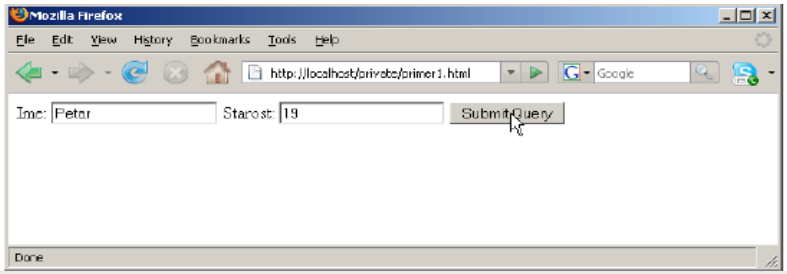
HTML dokument sa dva polja za unos podataka može da izgleda ovako:

```
<html>
<body>
<form action="prikaz.php" method="post">
Ime: <input type="text" name="ime" />
Starost: <input type="text" name="starost" />
<input type="submit" />
</form>
</body>
</html>
```

Dokument se snima kao npr. primer1.html. Ovaj dokument sadrži dva polja za unos teksta i jedno Submit Query dugme. Kada se u polja unesu odgovarajući podaci, oni se prosleđuju PHP skriptu koji se nalazi u datoteci prikaz.php. Skript može da izgleda ovako:

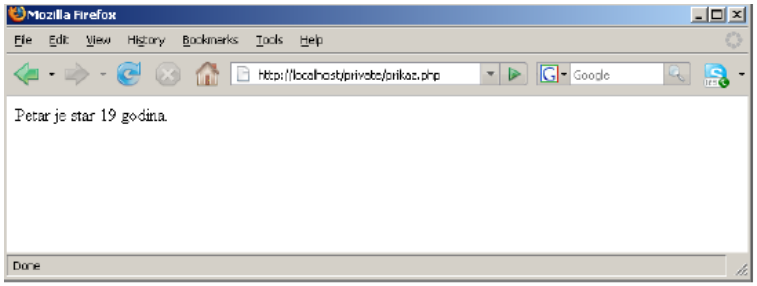
```
<?php echo $_POST["ime"]; ?> je star
<?php echo $_POST["starost"]; ?> godina
```

Nakon snimanja skripta i otvaranja HTML dokumenta uz pomoć npr. **phpdev** paketa, na već prikazani način, dobija se forma data na slici 1.



Slika-1 Dobijena forma

U zadata polja unose se parametri i kada se klikne na **Submit Query**, isti se prosleđuju PHP skriptu. Dobija se rezultat prikazan na slici 2.



Slika-2 Dobijen rezultat nakon što se klikne na Submit Query

PRIMER 1 – PREUZIMANJE I VALIDACIJA PODATAKA

*Datoteka koristi promenljivu **$_POST** za preuzimanje podataka*

Prilikom prosleđivanja, URL ne sadrži nikakve dodatne informacije i izgleda ovako:

http://localhost/private/prikaz.php

Nakon prosleđivanja, datoteka **prikaz.php** koristi promenljivu **$_POST** za preuzimanje podataka. Ova promenljiva predstavlja niz imena promenljivih i odgovarajućih vrednosti, prosleđenih uz pomoć HTTP POST metode. Ova promenljiva se koristi za prikupljanje vrednosti iz forme uz pomoć metode koja se u dokumentu definiše sa **method="post"**. Informacije koje se prosleđuju ovom metodom nisu vidljive za druge i ne postoje nikakva ograničenja vezana za npr. dužinu podataka koji se mogu proslediti.

Prikazani postupak često se koristi za validaciju podataka unetih u formu. Validaciju treba raditi uvek kada je to moguće. Validacija na klijentskoj strani je brža i smanjuje potrebu da se koristi server. Međutim, kod sajtova koji imaju veliki broj korisnika, gde je potrebno voditi računa o raspoloživim resursima servera, važno je razmišljati i o bezbednosti. Validaciju podataka na serverskoj strani treba koristiti uvek kada postoji potreba da se pristupi nekoj udaljenoj bazi podataka.

Dobar način za validaciju na serverskoj strani je da forma prosledi parametre samoj sebi, umesto prelaska na drugu stranu. U tom slučaju, korisnik će dobiti poruku o grešci na istoj strani na kojoj se nalazi i forma. To olakšava pronalaženje grešaka.

PRIMER 1 – GET METODA

Osim promenljive $_POST, za prosleđivanje parametara koristi se i promenljiva $_GET, koja koristi odgovarajući GET metod

Osim promenljive **$_POST**, za prosleđivanje parametara koristi se i promenljiva **$_GET**, koja koristi odgovarajući GET metod. Promenljiva **$_GET** predstavlja skup imena promenljivih i vrednosti koje se šalju uz pomoć HTTP GET metode. Informacije koje se šalju iz forme uz pomoć GET metode su vidljive za svakog, odnosno prikazaće se u adresnom prostoru čitača. Postoji i ograničenje vezano za podatke koji se mogu proslediti, tako da je dužina ograničena na 100 karaktera.

Ako bismo formu iz prvog primera promenili tako da sada izgleda ovako:

```
<form action="welcome.php" method="get">
Ime: <input type="text" name="ime" />
Starost: <input type="text" name="starost" />
<input type="submit" />
</form>
```

kada se klikne na Submit Query dugme, URL koji se prosleđuje izgledaće ovako:

http://localhost/private/prikaz.php?ime=Petar&starost=19

U tom slučaju PHP skript treba promeniti tako da koristi **$_GET** promenljivu:

<?php echo $_GET["ime"]; ?> je star
<?php echo $_GET["starost"]; ?> godina.

Imajući u vidu da se imena promenljivih i njihove vrednosti vide u okviru URL, GET metod ne bi trebalo koristiti za prosleđivanje šifara ili drugih važnih poverljivih informacija. Sa druge strane, na ovaj način omogućen je tzv. *bookmark* date strane, što je ponekad veoma korisno.

Osim navedenih, koristi se i **$_REQUEST** koja sadrži vrednosti promenljivih **$_POST**, **$_GET** i **$_COOKIE**, tako da se ova promenljiva koristi za dobijanje podataka prosleđenih i uz pomoć GET i POST metoda.

Često se prilikom korišćenja PHP jezika javlja potreba da se koristi više sličnih promenljivih. Umesto uvođenja različitih naziva za promenljive, što može biti problem naročito ako je njihov broj veliki, koriste se nizovi. Svaki element u nizu ima svoj ID (indeks), tako da mu se može lako pristupiti.

Nizovi se dele u tri kategorije:

- numerički nizovi sa numeričkim ID oznakama,
- asocijativni nizovi gde je svakom ID ključu pridružena neka vrednost, i
- višedimenzionalni nizovi koji sadrže jedan ili više nizova.

Promenljivama u numeričkom nizu vrednosti se mogu dodeliti na različite načine..

PRIMER 2

Napisti PHP skript određivanja prisustva studenata na predavanju

Zadatak:

Neka je **$student** niz sastavljen od imena studenata: Petar, Kosta, Jovan i Dragan. Napisati PHP skript koji će prikazati tekst da prvi i treći student nisu na času dok su ostali prisutni.

Rešenje:

Indeksi kod nizova obično počinju od nule, pa prvi student u nizu ima indeks 0, drugi ima indeks 1 itd. PHP skript može da izgleda ovako:

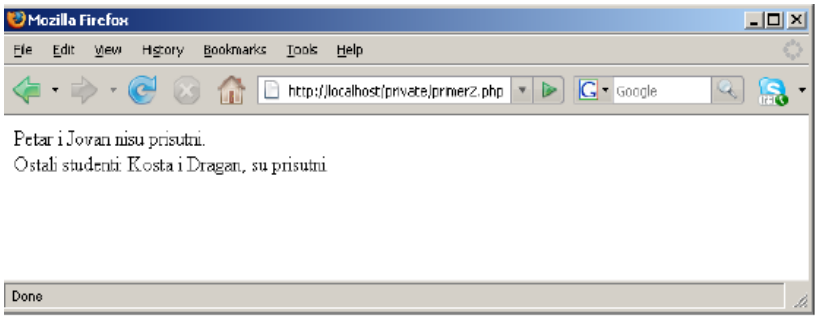
```
<?php
$student[0] = "Petar";
$student[1] = "Kosta";
$student[2] = "Jovan";
$student[3] = "Dragan";
echo $student[0] . " i " . $student[2] . " nisu prisutni.<br />
Ostali studenti: ". $student[1] . " i " . $student[3] . ", su prisutni.";
?>
```

Nakon snimanja kao npr. “primer2.php“ u čitaču će se dobiti rezultat dat na slici 3.

U PHP jeziku postoji i mogućnost automatskog dodeljivanja indeksa. Za poslednji primer, kreiranje niza moglo je da se izvede i na sledeći način:

$student = array("Petar", "Kosta", "Jovan", "Dragan");

Navedeni studenti imali bi isti indeks u nizu **$student**. Kada je potrebno uz pomoć niza zapamtiti neke specifične vrednosti za određene promenljive, koristi se asocijativni niz. Kod asocijativnih nizova, vrednosti indeksa definiše korisnik.



Slika-3 Rezultat

PRIMER 3

Kreirati niz $smer koji sadrži imena smerova

Zadatak:

Kreirati asocijativni niz **$smer** koji sadrži nazive smerova koje su upisali studenti, gde se kao ID pojavljuje ime studenta.

Rešenje:

U ovom slučaju, niz se koristi da se različitim studentima dodele odgovarajući smerovi na sledeći način:

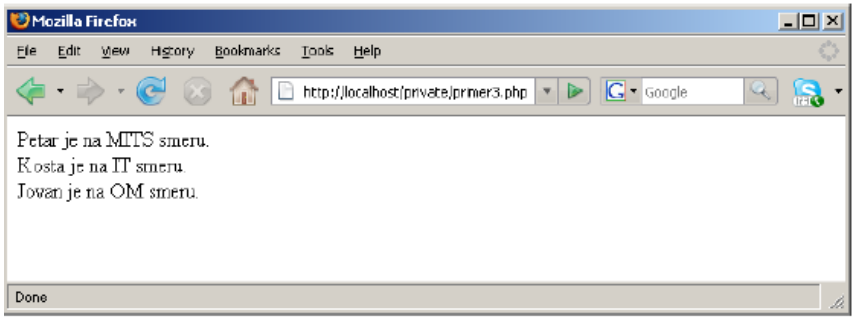
```
<?php
$smer['Petar'] = "MITS";
$smer['Kosta'] = "IT";
$smer['Jovan'] = "OM";
echo "Petar je na " . $smer['Petar'] . " smeru.<br />";
echo "Kosta je na " . $smer['Kosta'] . " smeru.<br />";
echo "Jovan je na " . $smer['Jovan'] . " smeru.";
?>
```

Ako skript snimimo kao "primer3.php", prikaz u čitaču izgleda kao na slici 4.

Slično kao i u drugom primeru, ovaj asocijativni niz se može kreirati na sledeći način:

$smer = array("Petar"=>MITS, "Kosta"=>IT, "Jovan"=>OM);

U nastavku će biti prikazano još nekoliko primera korišćenja PHP jezika.



Slika-4 Rezultat prikazan u veb čitaču

PRIMER 4

Kreirati PHP skript koji ispusuje datum u crvenoj boji

Zadatak:

Kreirati PHP skript koji će u čitaču prikazati trenutni datum ispisan crvenom bojom, veličine 24.

Rešenje:

Funkcija **date()** omogućava preuzimanje trenutnog vremena na serveru i njegovo formatiranje u skladu sa potrebama korisnika. U zavisnosti od parametara funkcije, način prikazivanja datuma može biti različit.

Parametri mogu biti sledeći:

- **d** – dan (ima vrednost od 01 do 31),
- **m** – mesec (ima vrednost od 01 do 12),
- **Y** – tekuća godina.

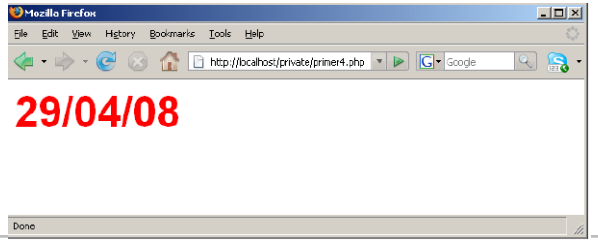
Osim toga, kao parametri mogu se pojaviti i sledeći karakteri: "/", "." ili "-", koji služe za razdvajanje vrednosti za dan, mesec i godinu, tako da se funkcija može koristiti na jedan od sledećih načina:

- date("Y/m/d"),
- date("Y.m.d"),
- date("Y-m-d").

Osim toga, korisnik može da definiše način prikazivanja datuma uz pomoć datih parametara, pa ako je npr. potrebno da datum bude ispisan u formatu **dan/mesec/godina**, funkciju **date()** treba da koristi na sledeći način: **date("d/m/y")**.
U ovom slučaju HTML dokument i odgovarajući skript mogu da izgledaju ovako:

```
<html>
<head>
</head>
<body>
<font face="Arial" color="#FF0000" size="24"><strong>
<?php print(date("d/m/y")); ?>
</strong></font>
</body>
</html>
```

Nakon snimanja kao **"primer4.php"** i otvaranja, prikaz u čitaču može da izgleda kao na slici 5.



PRIMER 5

Kreirati PHP skript za generisanje slučajnog broja

Zadatak:

Kreirati PHP skript za generisanje slučajnog broja u domenu od 1 do 5, promenu boje pozadine u zavisnosti od dobijenog broja.

Ako je dobijeni broj jednak 1, boja pozadine treba da bude plava, za broj 2 zelena, broj 3 crvena, za broj 4 žuta i za dobijeni broj 5 bela.

Rešenje:

Za generisanje slučajnog broja koristi se **rand()** funkcija u sledećem obliku:

rand(min,max);

gde su **min** i **max**, odgovarajuća minimalna i maksimalna vrednost koju funkcija **rand()** može da generiše. U ovom slučaju koristićemo: **rand(1,5)**. Ovaj broj će se generisati pri svakom učitavanju ili ažuriranju strane i taj broj dodeljujemo promenljivoj **$broj**.

Za boju pozadine uvedena je promenljiva **$pozadina**, čija je vrednost odgovarajući heksa-decimalni broj koji zavisi od vrednosti promenljive **$broj**. Tražene heksa-decimalne vrednosti su: #0000FF za plavu boju, #008000 za zelenu, #FF0000 za crvenu, #FFFF00 za žutu, i #FFFFFF za belu boju.

Za dodeljivanje odgovarajuće vrednosti promenljivoj $pozadina koristićemo if...else i elseif naredbe, o kojima je već bilo reči.

Da bi dobijena vrednost mogla da se koristi kao vrednost atributa bgcolor u HTML dokumentu, koristi se print naredba koja HTML deo izdvaja od PHP skripta.

Skript može da izgleda ovako:

```
<html>
<head>
</head>
<?
$broj = rand(1,5);
print("Slučajni broj je: $broj");
if($broj == 1)
{
$pozadina = "#0000FF";
}
elseif($broj == 2)
{
$pozadina = "#008000";
}
elseif($broj == "3")
{
$pozadina = "#FF0000";
}
elseif($broj == "4")
{
$pozadina = "#FFFF00";
```