Web dizajn i programiranje

Prof. dr.sc. Dragutin Kermek
Sveučilište u Zagrebu
Fakultet organizacije i informatike
Pavlinska 2, Varaždin 42000
dkermek@foi.hr

15. dio

Programiranje na strani poslužitelja

Uvod u programiranje na strani poslužitelja/CGI.
Programski jezik PHP.
Izvršavanje obrade (pretraživanje, zapis podataka i sl.) na strani poslužitelja.
Ponovno korištenje koda.
Rad s datumom i vremenom. Objektna orijentacija.
Slanje e-mail poruke. Generiranje HTML stranica.
Upravljenje pogreškama.
Korištenje i administriranje baze podataka (MySQL).
Realizacija autentikacije i autorizacije korisnika.

Što je PHP

PHP je skriptni jezik za programiranje na strani poslužitelja. Za njegovo izvođenje potreban je Web poslužitelj (Apache, IIS i dr.) s ugrađenom instalacijom PHP modula ili kao CGI binary.

Kratica: “PHP: Hypertext Preprocessor"

Sintaksa PHP sliči jezicima: C, Java i Perl.

PHP programski kod ugrađuje se u tekst HTML dokumenta.

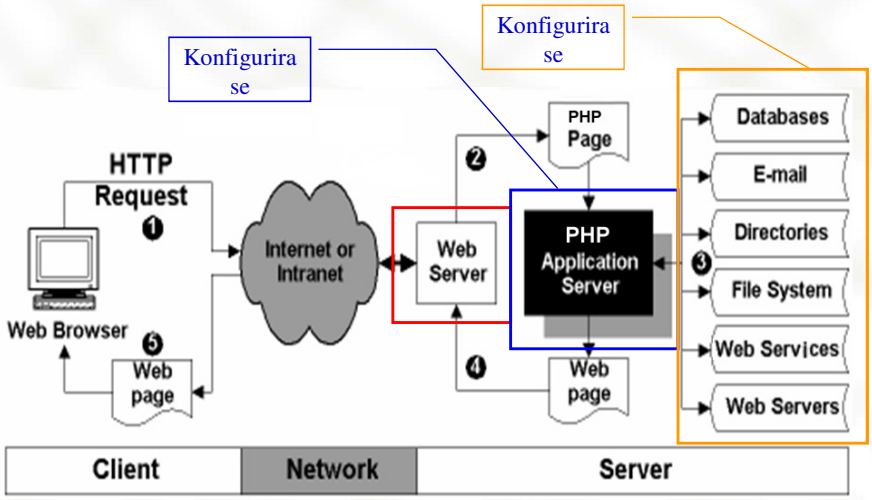
Svaki programski jezik određen je svojom sintaksom.

Bogatstvo programskog jezika izražava se brojem i raznolikošću biblioteka funkcija ili klasa koje stoje na raspolaganju (besplatno ili se kupuje).

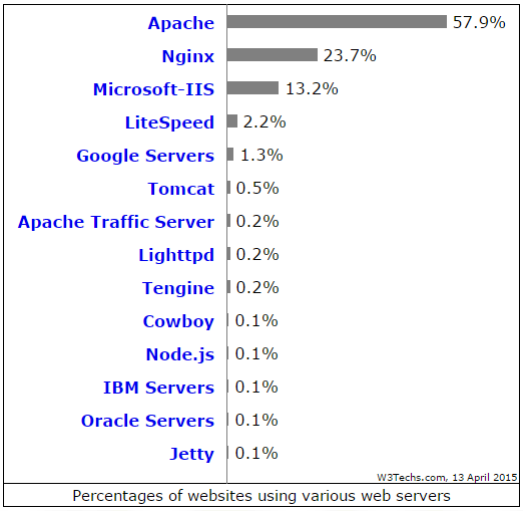
Što može PHP

- Sve što i bilo koji drugi CGI program
  - Preuzimati podatke iz formi
  - Generirati dinamičke web stranice
  - Slati i primati cookie
  - Rad sa bazama podataka (Adabas D, Dbase, FrontBase, Solid, Empress, mSQL, Sybase, FilePro (read-only), Direct MS-SQL, Velocis, IBM DB2, MySQL, Unix dbm, Informix, ODBC, Ingres, Oracle)
  - Podržava i druge protokole: IMAP, SNMP, NNTP, POP3, HTTP, LDAP
  - Moguće je korištenje XML (SAX i DOM standardi su podržani) za Web servise, RSS
  - Generiranje PDF i drugih formata

Što je potrebno za PHP



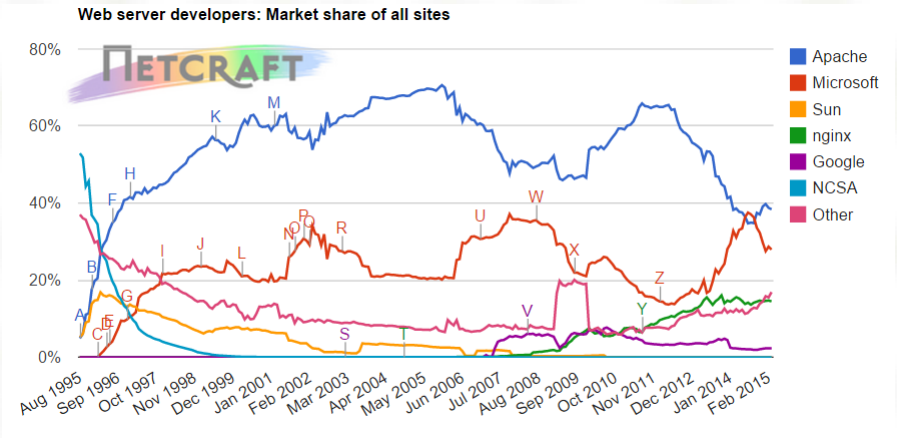
Statistika web poslužitelja



Percentages of websites using various web servers

http://w3techs.com/technologies/overview/web_server/all

Statistika web poslužitelja



http://news.netcraft.com/

Statistika web poslužitelja

Developer	February 2015	Percent	March 2015	Percent	Change
Apache	342,480,920	38.77%	337,175,536	38.39%	-0.38
Microsoft	253,484,221	28.69%	245,496,533	27.95%	-0.74
nginx	130,093,899	14.73%	127,191,696	14.48%	-0.25
Google	20,238,057	2.29%	20,097,702	2.29%	-0.00

http://news.netcraft.com/

Povijest PHP-a

Autor: **Rasmus Lerdorf**, jesen 1994

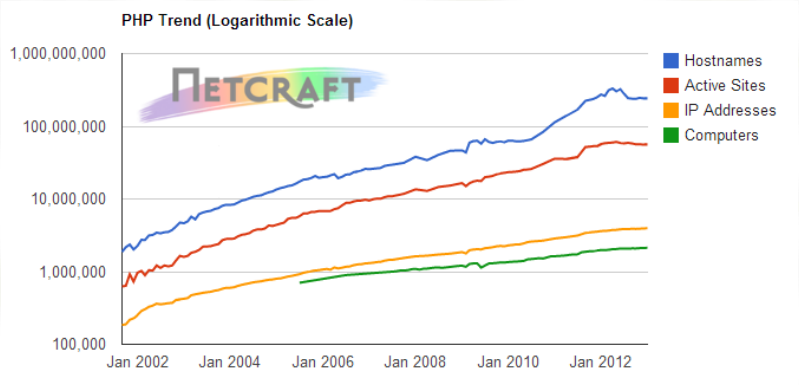
Prva verzija koja je bila dostupna za korištenje izašla je negdje početkom 1995 pod imenom "Personal Home Page Tools".

Krajem 1996 odprilike 15,000 web mjesta je koristilo PHP, a sredinom 1997 više od 50,000. Krajem 1999 procjenjeno je da više od 1,000,000 web mjesta koristi PHP.

PHP, http://www.php.net/

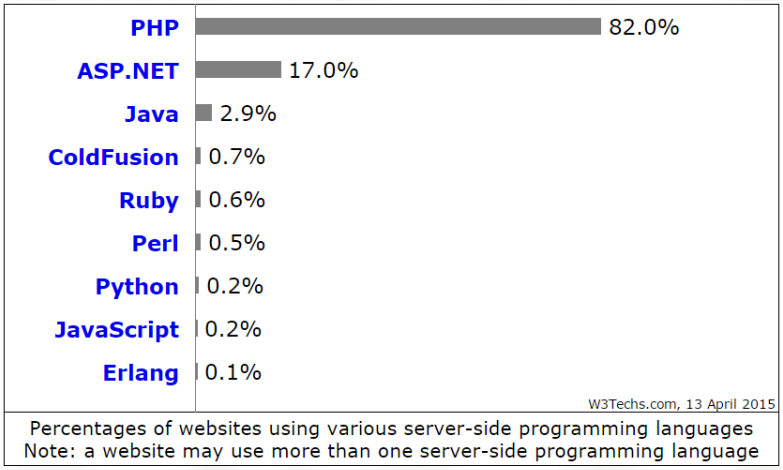
Rasmus Lerdorf, http://www.lerdorf.com/

Statistika PHP-a



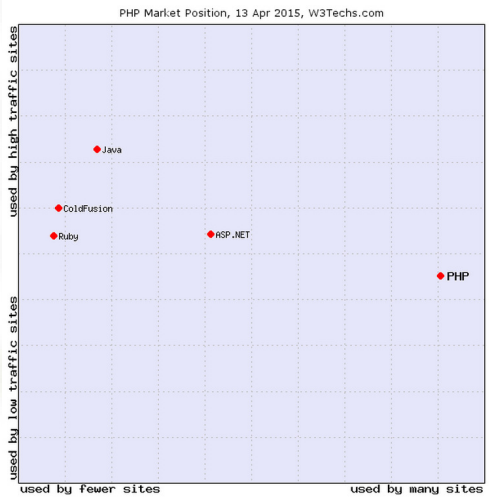
http://php.net/usage.php

Statistika PHP-a



http://w3techs.com/technologies/overview/programming_language/all

Statistika PHP-a



http://w3techs.com/technologies/details/pl-php/all/all

Statistika PHP-a

- **Popularna web mjesta koja koriste PHP:**
  - Facebook.com
  - Baidu.com
  - Wikipedia.org
  - Twitter.com
  - Taobao.com
  - Qq.com
  - Sina.com.cn
  - Tmall.com
  - Vk.com
  - Sohu.com

http://w3techs.com/technologies/details/pl-php/all/all

PHP i FOI

Rasmus Lerdorf, 19. travanj 2006. predavanje na FOI u sklopu suradnje s Danima otvorenih računarskih sustava u Zagrebu (http://www.open.hr/dc2006/)
http://old.foi.hr/novosti_informacije/2006/04/predavanja_19.html

PHP Presentation System, http://talks.php.net/index.php/PHP
Web 2.0 and PHP 5, http://talks.php.net/show/varazdin

PHP, Lerdorf i FOI

PHP verzije

PHP 1.0 – 1995.
PHP/FI 2.0 – 1997. (FI - Forms Interpreter)
PHP 3.0 – 6. 1998.
PHP 4.0 – 5. 2000.
PHP 5.0 – 7. 2004.
PHP 5.3.5 i 5.2.17 – 6.1.2011. - važeće verzije
PHP 5.4.0 i 5.3.10 - 12.3.2012. - važeće verzije
PHP 5.4.14 i 5.3.24 - 21.4.2013. - važeće verzije
PHP 5.5.11, 5.4.27 i 5.3.28 - 28.4.2014. - važeće verzije
PHP 5.6.7, 5.5.23 i 5.4.39 - 13.4.2015. - važeće verzije
PHP 7.0.5, PHP 5.6.20 i 5.5.34 - 11.4.2016. - važeće verzije
PHP 7.1.3, PHP 7.0.17 i 5.6.30 - 09.4.2017. - važeće verzije

PHP primjer

```
<html>
<head>
<title>PHP - Primjer 00</title>
</head>
<body> <?php echo "Opa, ovo je PHP!"; ?> </body>
</html>
```

Korisnik nakon izvođenja prima čisti HTML

```
<html>
<head>
<title>PHP - Primjer 00</title>
</head>
<body> Opa, ovo je PHP! </body>
</html>
```

Primjer00 – Ispis poruke

PHP primjer - ispis postavki

```
<html>
<head>
<title>PHP - Primjer 01</title>
</head>
<body> <?php phpinfo(); ?> </body>
</html>
```

Primjer01 – Ispis postavki PHP-a

PHP primjer - ispis postavki

PHP Version 5.4.7

System	Windows NT NBKERMEK4 6.2 build 9200 (Unknown Windows version Enterprise Edition) i586
Build Date	Sep 12 2012 23:44:56
Compiler	MSVC9 (Visual C++ 2008)
Architecture	x86
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--disable-isapi" "--enable-debug-pack" "--without-mssql" "--without-pdo-mssql" "--without-pi3web" "--with-pdo-oci=C:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=C:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8-11g=C:\php-sdk\oracle\instantclient11\sdk,shared" "--enable-object-out-dir=../obj/" "--enable-com-dotnet=shared" "--with-mcrypt=static" "--disable-static-analyze" "--with-pgo"
Server API	Apache 2.4 Handler Apache Lounge
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\Windows
Loaded Configuration File	E:\servisi\xampp\php\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20100412

Sintaksa jezika / 1.

Izlazak iz HTML-a i ulaz u "PHP kod".
Postoje nekoliko načina prijelaza iz HTML-a u PHP:

1. `<?php echo("1. način \n"); ?>` – standardan način

2. `<? echo ("2. način \n"); ?>` – ovisi o konfiguraciji PHP-a

Datoteka PHP postavki (**php.ini**) sadrži postavku kojom se određuje prijelaz iz HTML-a u PHP:

```
short_open_tag = On
```

Primjer 1 na predavanjima: XAMP – Off - port 80 (PHP 7.0.9)
Primjer 2 na predavanjima: WAMP - On - port 9090 (PHP 5.6.25)

Primjer02 – Sintaksa

Sintaksa jezika

Instrukcije su odvojene kao u jezicima C i Perl (točka-zarez):

```
<?php echo "1. način"; ?>

          ILI
<?php
    echo " 2. način ";
?>
```

Primjer03 – Ispis poruka

Varijable

Imena varijabli se mogu kreirati po sljedećem pravilu:
$[a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*
PHP, kao i drugi skriptni jezici, NE zahtjeva deklariranje tipa podatka.
Varijablu možemo deklarirati pomoću naziva bez pridruživanja vrijednosti, koja u tom slučaju nema pridruženi tip i tretira se kao prazna varijabla.

```
<?php
$a = 1;
$b = 7;
$sesija = ...
$_brojRedaka = ...

$_kolicina;
...
?>
```


Sintaksa jezika / 2.

Postoji razlika između jednostrukih i dvostrukih navodnika!
Jednostruki navodnici doslovno prikazuju tekst dok dvostruki provode zamjenu varijabli s njihovim vrijednostima.

```
$a = 1;
$b = 7;

<? echo "$a + $b " . ($a + $b) . "\n"; ?> <br>
<? echo '$a + $b ' . ($a + $b) . '\n'; ?> <br>
```

Primjer04 – Primjena navodnika

Komentari

Php koristi 'C', 'C++' i 'Unix shell' tipove komentara.

```
<?php
   echo "Ovo je komentar"; // c++ komentar
?>
   <br>
<?php
   /* višelinijski komentar u stilu C-a */
   echo "Ovo je kraj komentara"; # unix shell komentar
?>
```

Nije dozvoljeno ugnjezditi višelinijske komentare npr.

```
   /* ovo je u redu /* ovo nije u redu */ */
```

Primjer05 – Komentari

Ponovno korištenje koda

Razvoj PHP aplikacija može biti jednostavnije i brže ako se provodi strukturiranje sadržaja Web stranice s ciljem stvaranje predložaka stranica. Rezultat toga je izdvajanje pojedinih dijelova koji se ponavljaju u svim stranicama (zaglavlja, podnožja, izbornici i sl) u zasebne datoteke koje se uključuju u kod pojedinačnih stranica.

To se može provreti na 2 načina:

```
include(nazivdatoteke);  include_once(nazivdatoteke);
require(nazivdatoteke);  require_once(nazivdatoteke);
```

Između njih postoji malo razlika jer `require` generira fatalnu pogrešku kada datoteka ne postoji dok `include` samo upozorenje.

Ponovno korištenje koda / 1.

```
<?php
include('zaglavlje.inc');
?>
Ovo je tekst!!!
<?php
include('podnozje.inc');
?>
```

```
<html>
<head>
<title>PHP - Primjer 39</title>
</head>
<body>
<p>Ovo je zaglavlje</p>
```

```
<p>Ovo je podnožje</p>
</body>
</html>
```

Primjer39 – Ponovno korištenje koda

Ponovno korištenje koda / 2.

```
<?php
require('zaglavlje.inc');
?>
Ovo je tekst!!!
<?php
require('podnozje.inc');
?>
```

```
<html>
<head>
<title>PHP - Primjer 39</title>
</head>
<body>
<p>Ovo je zaglavlje</p>
```

```
<p>Ovo je podnožje</p>
</body>
</html>
```

Primjer39_1 – Ponovno korištenje koda

Ponovno korištenje koda / 3.

```
<?php
include('zaglavlje.inc');
include('zaglavlje.inc');
?>
Ovo je tekst!!!
<?php
include('podnozje.inc');
include_once('podnozje.inc');
?>
```

```
<html>
<head>
<title>PHP - Primjer 39</title>
</head>
<body>
<p>Ovo je zaglavlje</p>
```

```
<p>Ovo je podnožje</p>
</body>
</html>
```

Primjer39_2 – Ponovno korištenje koda

Tipovi podataka

PHP podržava sljedeće tipove podataka:

- Arrays – nizovi (indeksirani i asocijativni)
- Floating-point numbers – decimalni brojevi
- Integer - cjelobrojni
- Object - objekt
- String - znakovni

Integer

Integer možemo specificirati na jedan od sljedećih načina:

```
$a = 1234; # decimalni broj
$a = -123; # negativan broj
$a = 0123; # oktalni broj (83 decimalno)
$a = 0x12; # heksadecimalni broj (18 decimalno)
```

Veličina integer-a zavisi od platforme na kojoj se izvodi PHP, iako je obično 32-bitan.

Floating point – decimalni tip

Sintaksa:

```
$a = 1.234;
$a = 1.2e3;
```


Polja/nizovi

Nizovi se u PHP-u ponašaju:

- kao vektori odnosno indeksirani nizovi (redni broj, vrijednost)
- kao hash tablice (rječnik) odnosno asocijativni nizovi (par ključ, vrijednost)

Prepoznavaju se po `[]` nakon naziva varijable:

```
$zapis[0] = 123;
$zapis[1] = "dkermek";

$podaci["broj"] = 123;
$podaci["korisnik"] = "dkermek";
```


Jednodimenzionalni nizovi

Niz možemo kreirati koristeći array() funkciju ili možemo eksplicitno pridružiti vrijednost svakom elementu niza

```
$a[0] = "abc";
$a[1] = "def";
$b["foo"] = 13;
```

Možemo koristiti i prazne zagrade pa će element biti automatski dodan na kraj zagrade

```
$a[] = "hello"; // $a[2] == "hello"
$a[] = "world"; // $a[3] == "world"
```


Jednodimenzionalni nizovi / 1.

Funkcije za sortiranje i ispis nizova:

- **asort()** – sortira niz i održava redoslijed pridruživanja
- **arsort()** – sortira niz u obrnutom redoslijedu i zadržava redoslijed pridruživanja
- **ksort()** – sortira niz po ključu
- **rsort()** – sortira niz u obrnutom redoslijedu (od najvećeg prema najmanjem
- **sort()** – sortira niz od najmanjeg prema najvećem elementu
- **uasort()**, **usort()**, **uksort()** - sortiraju nizove po funkcijama koje im se zadaju
- **print_r()** – ispisuje informaciju u varijabli (ne samo za niz) u čitljivom obliku
- **var_dump()** – ispisuje informaciju o varijabli (ne samo za niz)

Višedimenzionalni nizovi / 1.

Za svaku novu dimenziju dodamo novi [indeks] ili [ključ].

```
$a[1][0] = $f; # dvodmenzionalni niz
$a["foo"][2] = $f; # (     možemo miješati numeričke i)
$a[3]["bar"] = $f; # (     asocijativne ključeve)
$a["foo"][4]["bar"][0] = $f; # četiri dimenzionalni niz!
```

Višedimenzionalne nizove ne možemo direktno referencirati unutra stringova

```
echo "Ovo neće raditi: $a[3][bar]";

echo " Ovo hoće raditi : " . $a[3][bar];        // PHP3
echo " Ovo hoće raditi : {$a[3][bar]}";         // PHP4
```


Višedimenzionalni nizovi / 2.

Nizove možemo puniti na slijedeće načine:

Prvi način:

```
$a["color"] = "red";
$a["taste"] = "sweet";
$a["name"] = "apple";
```

Drugi način:

```
$a = array(   "color" => "red",
              "taste" => "sweet",
              "name" => "apple");
```


Višedimenzionalni nizovi / 3.

Funkcija **array()** može biti ugnježđena kod višedimenzionalnih nizova:

```
<? $a = array(
   "more" => array("boja" => "plava", "okus" => "slano"),
   "jezero" => array("boja" => "zelena", "okus" => "slatko"),
   "bara" => array( "boja" => "smeđa", "okus" => "bljutavo"));

   echo $a["more"]["boja"];
   echo '<pre>';
   print_r($a);
   echo '</pre>';
?>
```

Primjer06 – Višedimenzionalni nizovi

Objekti

Da bi instancirali klasu (kreirali njen objekt) u varijablu koristi se operator **new**. Kasnije se obrađuju klase:

```
<?php
        class proba {
                function probaj() {
                        echo "Probam.";
                }
        }

$bar = new proba;
$bar->probaj();
?>
```

Primjer07 – Objekti

Rukovanje tipovima podataka

PHP, kao i drugi skriptni jezici, NE zahtjeva deklariranje tipa podatka. Postupak se provodi automatski na bazi vrste operatora tako da se uvijek pretvara u viši tip podatka:

```
$foo = "0"; // $foo je string (ASCII 48)
$foo++; // $foo je string "1" (ASCII 49)
$foo += 1; // $foo je integer (2)
$foo = $foo + 1.3; // $foo sada je double (3.3)
$foo = 5 + "10 Little Piggies"; // $foo je integer (15)
$foo = 5 + "10 Small Pigs"; // $foo je integer (15)
$foo = 5 + "Small Pigs"; // $foo je integer (5)
$foo = "Small Pigs" + 5; // $foo je integer (5)
$foo = 5 . "Small Pigs"; // $foo je string ("5Small Pigs")
$foo = "Small Pigs" . 5; // $foo je string ("Small Pigs5")
```

Primjer08 – Tipovi podataka

Rukovanje tipovima podataka / 1.

Još malo primjera:

```
$foo = 1 + "10.5"; // $foo is double (11.5)
$foo = 1 + "-1.3e3"; // $foo is double (-1299)
$foo = 1 + "bob-1.3e3"; // $foo is integer (1)
$foo = 1 + "bob3"; // $foo is integer (1)
$foo = 1 + "10 Small Pigs"; // $foo is integer (11)
$foo = 1 + "10 Little Piggies"; // $foo is integer (11)
$foo = 1 + "10.7 Small Pigs"; // $foo is double (11.7)
$foo = 1 + "10.7 Little Piggies"; // $foo is double (11.7)
$foo = "10.7 pigs " + 1; // $foo is double (11.7)
$foo = "10.7 pigs " + 1.0; // $foo is double (11.7)
```

Primjer09 – Rukovanje tipovima podataka

Pretvaranje tipova podataka

Ako se strogo želi koristiti određeni tip podataka to se čini funkcijom

```
int settype (string var, string type)
```

Pretvara varijablu 'var' u tip podataka 'type', koji može imati sljedeće vrijednosti:

```
"integer",   "double",
"string",    "array",     "object"
```

Funkcija vraća '`true`' ako je promjena uspješna u suprotnom vraća '`false`'.

Suprotno radi funkcija: `string gettype (mixed var)`

Pretvaranje tipova podataka / 1.

```
<?
$foo = 1 + "10.5"; // $foo is double (11.5)
print $foo . " " . gettype($foo) . "<br>";
settype($foo, "string");
print $foo . " " . gettype($foo) . "<br>";
settype($foo, "int");
print $foo . " " . gettype($foo) . "<br>";
?>
```

Primjer10 – Pretvaranje tipova podataka

Testiranje tipa podatka

Ako se strogo želi ispitati određeni tip podataka to se čini funkcijom

```
is_array()
is_double(), is_float(), is_real()
is_long, is_int(), is_integer()
is_string()
is_object()
```


Testiranje tipa podatka / 1.

```
<?
$foo = 10.5;
echo "int? " . is_int($foo) . "<br>";
echo "double? " . is_double($foo) . "<br>";
echo "string? " . is_string($foo) . "<br>";
?>
```

Primjer10_1 – Testiranje tipa podatka

Testiranje statusa varijable

Provjera postojanja varijable, njeno brisanje i imali li vrijednost:

isset()

unset()

empty()

- "" (pazan string)
- 0 (0 kao broj)
- "0" (0 kao string)
- NULL
- FALSE
- array() (prazno polje)
- $var; (deklarirana varijabla bez vrijednosti)

Testiranje statusa varijable / 1.

```
//$boo = 0;
$foo = $boo + 4;
$foo = 10.5;
echo "postoji? " . isset($foo) . "<br>";
echo "prazna? " . empty($foo) . "<br>";
unset($foo);
echo "postoji? " . isset($foo) . "<br>";
echo "prazna? " . empty($foo) . "<br>";
$pero = "2";
echo "pero: prazna? " . empty($pero) . "<br>";
$pero = false;
echo "pero: postoji? " . isset($pero) . "<br>";
echo "pero: prazna? " . empty($pero) . "<br>";
```

Koristi se varijabla koja nije definirana

Varijabla se briše

Testiranje statusa varijable / 1.

```
$kol;
echo "kol: postoji? " . isset($kol) . "<br>";
echo "kol: prazna? " . empty($kol) . "<br>";
$kol = 1;
echo "kol: postoji? " . isset($kol) . "<br>";
?>
```

Varijabla je deklarirana, nema pridruženi tip podatka i nema pridruženu vrijednosti

Primjer10_2 – Testiranje statusa varijable

String

Označavamo ih navodnicima. Ako koristimo obične navodnike (") varijable unutar stringa će biti zamijenjene pravim vrijednostima. Drugi način za označavanje stringova su jednostruki navodnici ('). Unutar jednostrukih navodnika varijable se ne zamjenjuju svojim vrijednostima. String se može tretirati kao indeksirani niz znakova. Za specijalne znakove koristi se \ (backslash):

Slijed	**Značenje**
\n	linefeed (LF or 0x0A in ASCII)
\r	carriage return (CR or 0x0D in ASCII)
\t	horizontal tab (HT or 0x09 in ASCII)
\\	backslash
\$	dollar sign
\“	double-quote
\[0-7]{1,3}	znak u oktalnaj notaciji
\x[0-9A-Fa-f]{1,2}	znak u heksadecimalnoj notaciji

String / 1.

```
<?php
  $str = "Ovo je string";
  $str = $str . " sa još malo nadodanog teksta.";
  $str .= " I još malo teksta \n";
  print $str . "<br>";
  $num = 9;
  $str = "Broj: $num<br>";          // 'Broj: 9<br>'
  print $str;
  $num = 9;
  $str = 'Broj: $num<br>';          //  'Broj: $num<br>'
  print $str;
  $str = 'Ovo je test.';
  print $str . "<br>";
  $first = $str[0];                 // Uzimanje prvog
  $last = $str[strlen($str)-1];     // i zadnjeg znaka
  print $first . " " . $last;
?>
```

Primjer11 – String

String u više redova

Još jedan način je korištenje (<<<) oznake:

```
<?php
      $str = <<<EOD
            iza 3 znaka manje koristimo oznaku
            kojom označavamo
            početak i kraj stringa.
EOD;
?>
```

Primjer12 – String u više redova

Varijable i reference

```
<?php
   $foo = 'Bob';              // Pridružujemo vrijednost 'Bob'
   echo $foo . "<br>";
   $bar = &$foo;              // Referenciramo $foo preko $bar.
   $bar = "Pero";             // Mijenjamo $bar i $foo je
                              // promijenjen.
   echo $foo . "<br>";              // 'Pero '
   echo $bar . "<br>";              // 'Pero '
   $bar = "Moje ime je $bar";
   echo $foo . "<br>";              // 'Moje ime je Pero '
?>
```

Primjer13 – Varijable

Unaprijed definirane varijable

U PHP-u postoji veliki broj unaprijed definiranih varijabli.
Sljedeće varijable vezane su uz Apache server

Varijabla	Opis
SERVER_NAME	naziv servera
REQUEST_METHOD	koja metoda je korištena za pristup stranici; npr. 'GET', 'HEAD', 'POST', 'PUT'
SCRIPT_NAME	sadrži put do skripte koja se trenutno izvršava
REMOTE_ADDR	IP korisnika koji gleda stranicu
SERVER_PROTOCOL	ime i revizija protokola preko kojeg je dan zahtjev za stranicom npr. HTTP/1.0

Unaprijed definirane varijable / 1.

Varijabla	Opis
argv	niz argumenata proslijeđenih skripti
argc	ako je skripta pozvana iz komandne linije
PHP_SELF	ime skripte koja se izvršava
_COOKIE	asocijativni niz varijabli proslijeđenih skripti preko HTTP cookie-ja
_ENV	asocijativni niz varijabli koje se odnose na okolinu u kojoj se izvršava skripta
_FILES	asocijativni niz stavki koje su kreirane prilikom prenošenja datoteke tj. HTTP file upload
_GET	asocijativni niz varijabli proslijeđenih skripti preko HTTP GET metode
_POST	asocijativni niz varijabli proslijeđenih skripti preko HTTP POST metode

Posebne osobine varijabli

U PHP-u vrijednost varijable može i sama postati varijabla:

```
<?php
   $a = "Hello";
   $$a = "world";    // sadržaj varijable a je nova varijabla
                        pod nazivom $Hello kojoj se pridružuje
                        vrijednost "world"
   echo "$a <br>";          // ispisuje: Hello
   echo "$$a <br>";         // ispisuje: $Hello
   echo "${$a} <br>";       // ispisuje: world
   echo "$Hello <br>";      // ispisuje: world
   echo "$a ${$a} <br>";    // ispisuje: Hello world
   echo "$a $Hello";        // ispisuje: Hello world
?>
```

Primjer14 – Posebne osobine varijabli

Konstante

Konstante definiramo sa **`define()`** funkcijom. U PHP-u postoje i predefinirane konstante:

TRUE - istina
FALSE - laž
__FILE__ - ime skripte
__LINE__ - linija u skripti

PHP_VERSION PHP_OS
E_ERROR E_WARNING
E_PARSE E_NOTICE
E_ALL

Konstante / 1.

Primjer za definiciju konstante:

```
<?php
   define("POZDRAV", "Hello world.");
   echo POZDRAV;
?>
```

Primjer15 – Konstante

Konstante kontrole programa

Primjer za __FILE__ i __LINE__ :

```
<?php
   print "Ovo je: " . __FILE__ . " u liniji broj: " . __LINE__;
?>
```

Primjer16 – Konstante kontrole programa

Operatori

U PHP-u postoje sljedeće vrste operatora:

1. Aritmetičke operatore
2. Operatore pridruživanja
3. Operatore uspoređevanja
4. Operatore na razini bitova
5. Operatore za kontrolu grešaka
6. Operatore izvršavanja
7. Operatore za povećavanje i smanjivanje
8. Logičke operatore
9. Operatore za rad sa stringovima

Aritmetički operatori

+, -, /, *, %, ++, --

Operatori uspoređivanja

Primjer	Naziv	Rezultat
`$a == $b`	Jednako	True ako je $a jEdnak $b.
`$a === $b`	Identično	True ako je $a jednak $b i istog su tipa.
`$a != $b`	Nije jednako	True ako $a nije jednak $b.
`$a !== $b`	Nije identično	True ako $a nije jednak $b ili nisu istog tipa.
`$a < $b`	Manji od	True ako je $a strogo manji od $b.
`$a > $b`	Veći od	True ako je $a strogo veći od $b.
`$a <= $b`	Manji ili jednak	True ako je $a manje ili jednako $b.
`$a >= $b`	Veći ili jednak	True ako je $a veći ili jednak $b.

Logički operatori

Primjer	Naziv	Rezulat
`$a and $b`	I	True ako su $a i $b true. Ako je prvi false, drugi se ne ispituje
`$a or $b`	Ili	True ako je ili $a ili $b true. Ako je prvi true, drugi se ne ispituje.
`$a xor $b`	Eks. ili	True ako je ili $a ili $b true, ali ne oba.
`! $a`	Ne	True ako $a nije true.
`$a && $b`	I	True ako su $a i $b true.
`$a || $b`	Ili	True ako je ili $a ili $b true.

Logički operatori

A	B	==	!=	===	!==
2	„2”	1	0	0	1
2	2	1	0	1	0
2	3	0	1	0	1
2	„3”	0	1	0	1

A	B	A && B	A || B	!A	!(A && B)
0	0	0	0	1	1
0	1	0	1	1	1
1	0	0	1	0	1
1	1	1	1	0	0

Operatori tipa

PHP podržava samo jedan operatora tipa, a to je:

`instanceof`	instanca klase

String operatori

PHP podržava samo dva string operatora, a to su:

.	Operator spajanja stringova
.=	Operator pridruživanja stringova

if instrukcija

```
<?php
$a = 1;
$b = 2;
print `$a=` . $a . `, $b=` . $b . "<br>";
if ($a > $b) {
        print "a > b";
   }
   elseif ($a == $b){
        print "a == b";
   }
   else {
        print "a < b";
   }
?>
```

Primjer17 – if instrukcija

switch instrukcija

```
<?php
$i = 1;
switch ($i) {
   case 0: print "i = 0"; break;
   case 1: print "i = 1"; break;
   case 2: print "i = 2"; break;
        }
?>
```

Primjer18 – switch instrukcija

while instrukcija

```
<?php
$a = 0;
$b = 10;
while ($a < $b) {
       print "$a <br>";
        $a++;
   }
do {
       print "$a <br>";
        $a--;
   } while ($a > 0);
?>
```

Primjer19 – while, do-while instrukcija

for instrukcija

```
<?php
for ($i=0;$i < 10;$i++) {
       print "$i <br>";
        if($i == 7)
             break;
   }
?>
```

Primjer20 – for instrukcija

foreach instrukcija - za nizove

```
<?php
$arr = array('jedan', 'dva', 'tri');

foreach ($arr as $i) {
   echo "$i<br>\n";
   }
foreach ($arr as $j => $i) {
    echo "$j. $i<br>\n";
}
?>
```

Primjer21 – foreach instrukcija

foreach instrukcija / 1. - za nizove

```
<?php
$arr['jedan'] = 1;
$arr['dva'] = 2;
$arr['tri'] = 3;
foreach ($arr as $k => $vr)
   {
   echo "$k=$vr<br>\n";
   }
?>
```

Primjer22 – foreach instrukcija

Funkcije

Funkcije možemo definirati na sljedeći način:

- Bez argumenata:

```
function foo() {
        ...
}
```

- S argumentima (prijenos vrijednosti, bez promjene):

```
function foo ($arg_1, $arg_2, ..., $arg_n) {
        ...
}
```

- S argumentima (prijenos reference, može se promijeniti):

```
function foo (&$arg_1, &$arg_2, ..., &$arg_n) {
        ...
}
```

- S pretpostavljenim vrijednostima argumenata:

```
function foo ($arg_1 = pretpostavljena vrijednost) {
        ...
}
```


Funkcije / 1.

```
<?php
square(4);
square(40);
square(14);
square(24);
square(41);


function square ($num) { echo $num * $num; }
?>
```

Primjer23 – Funkcije

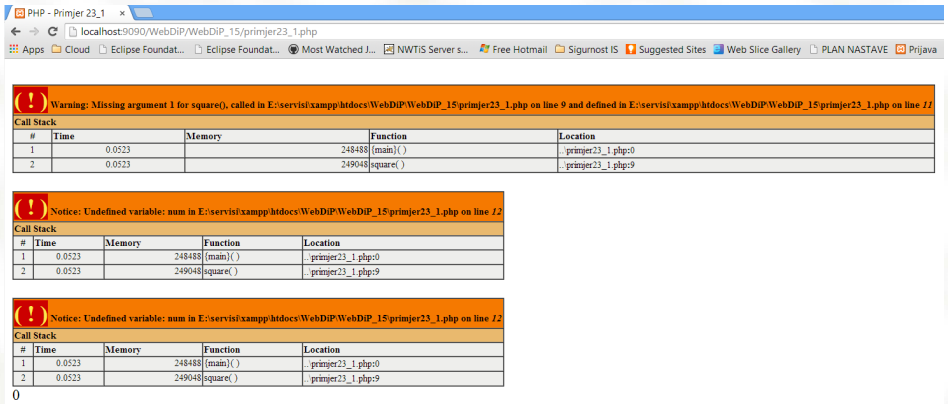
Funkcije / 2.

```
<?php
square();                    // nema argument

function square ($num) { echo $num * $num; }
?>
```

Primjer23_1 – Funkcije

Funkcije / 2.



U testnoj varijanti postavki javljaju se upozorenja.

Funkcije / 3.

```
<?php
square();                    // nema argument
echo "<br>";
square(3);

function square ($num = 7) { echo $num * $num; }
?>
```

Primjer23_2 – Funkcije

Funkcije / 4.

Kod funkcije kojoj nisu definirani argumenati, odnosno njihov broj može varirati:

```
function foo() {
    ...
}
```

broj argumenata i same argumente možemo dobiti preko posebnih funkcija:

func_num_args() // broj argumenata

func_get_arg() // preuzimanje argumenta

Funkcije / 5.

```
<?php
foo(1, "505", "pero", 2, 3, "IWA");

function foo()
{
   $numargs = func_num_args();
   echo "Broj argumenata: $numargs<br>\n";
   if ($numargs >= 2)
   {
        echo "2. argument je: " . func_get_arg(1) . "<br>\n";
   }
}
?>
```

Primjer23_3 – Funkcije

Vraćena vrijednost funkcije

Funkcije mogu vraćati:
Vrijednost:

```
function square ($num) { return $num * $num; }
```

Niz:

```
function small_numbers() { return array (0, 1, 2); }
```

Referencu:

```
function &returns_reference() { return $someref; }
```


Vraćena vrijednost funkcije / 1.

```
<?php
$a = 10;
echo "a=" . $a . " 4**2=" . square (4) . "<br>";

$b = small_numbers();
foreach ($b as $i) { echo $i . "<br>"; }

function square ($num) { return $num * $num; }

function small_numbers() { return array (0, 1, 2); }
?>
```

Primjer24 – Vraćene vrijednosti funkcije

Vraćena vrijednost funkcije / 2.

```
<?php
list ($zero, $one, $two) = small_numbers();
echo $zero . " " . $one . " " . $two . "<br>";

function small_numbers() { return array (0, 1, 2); }
?>
```

funkcijom list(...) varijablama iz liste argumenata pridružuju se korespondirajuće vrijednosti niza. Skraćuje se postupak jer zamjenjuje veći broj pridruživanja prema indesiranim pozicijama iz niza.

Primjer24_1 – Vraćene vrijednosti funkcije

Dinamički poziv funkcije

Dinamičko definiranje naziva fukcije koja se poziva dobije se tako da varijabla sadrži ime funkcije pa tu varijablu možemo koristiti da pozovemo funkciju:

```
<?php
function foo() { echo "foo()<br>\n";     }
function bar( $arg = '' ) {
      echo "bar(); argument je '$arg'.<br>\n";
}
$func = 'foo';
$func();
$func = 'bar';
$func('test');
?>
```

Primjer25 – Poziv funkcije

Objektna orijentacija u PHP-u

Veza prema objektnoj orijentaciji definirana je ulogama:

OO	**PHP**
klasa	**- klasa**
metoda	**- funkcija**
objekt	**- referenca prema nekom podatku unutar klase**

Moguća je realizacija većine OO osobina.

Klase

Programiranje u PHP-u može biti jednostavnije i brže ako se koriste gotove klase. Može se reći da su to biblioteke funkcija koje stoje na raspolaganju (ukoliko su instalirane na računalu).

Klasa se uključuje u skriptu

```
include('nazivKlase.php');
```


Vlastite klase

Realizacija većeg projekta zahtjeva drugačiji pristup nego što je slučaj kod malih projekata sa nekoliko skripata. Inženjerski pristup radu na projektu sigurno će biti usmjeren stvaranju vlastitih funkcija koje će se moći koristiti u više skripata. Te funkcije mogu se objediniti prema srodnosti u nekoliko klasa i tako višestruko koristiti u skriptama tog i kasnijih projekata.

Konstruktor mora imati isti naziv kao i klasa (<= 5.3.2). Od 5.3.3 obične metode.
Kontruktor može imati naziv __construct(...) (>= 5.3.3)
Destruktor ima naziv __destruct (void)

Vlastita klasa ima nastavak `.php` kao i svaka druga PHP skripta.

Vlastita klasa - Red - stari način (<=5.3.2)

```
<?
class Red
{
        public static $broj;
        public static $elementi;

        function Red() {
                $this->broj = 0;
                $this->elementi = array();
        }

        function broj() {
                return $this->broj;
        }

        function dodaj($noviElement) {
                $this->elementi[] = $noviElement;
                $this->broj++;
        }
```

naziv klase

varijable klase

kostruktor klase

funkcija klase

Vlastita klasa - Red / 1.

```
        function uzmi() {
                if($this->broj == 0) {
                        return null;
                }
                $odlazi = array_shift($this->elementi);
                $this->broj--;
                return $odlazi;
        }

        function elementi() {
                return $this->elementi;
        }
}
?>
```


Vlastita klasa - Red / 3.

```
<? include('Red.php');
$red = new Red;
$red->dodaj("Jedan");
$red->dodaj("Dva");
$red->dodaj("Tri");
$red->dodaj("Pet");
brojElemenata($red->broj());
ispisiElemente($red->elementi());
$elem = $red->uzmi();
print "Uzet element: $elem <br>";
brojElemenata($red->broj());
ispisiElemente($red->elementi());

function brojElemenata($broj) {
        print "Ukupno elemenata: $broj<br>";
}

function ispisiElemente($lista) {
        foreach ($lista as $i) {
                print $i . "<br>";
        }
}
```

Primjer40 – Vlastita klasa - Red

Vlastita klasa – Red / 4

```
<? // Red_1.php
class Red
{
        public static $broj;
        public static $elementi;

        function __contruct() {
                $this->broj = 0;
                $this->elementi = array();
        }

        function __destruct() {
                print "<br><b>Brišem elemente reda!</b><br>";
                while ($this->broj != 0) {
                        print $this->uzmi() . "<br>";
                }
        }

        function broj() {
                return $this->broj;
        }

        function dodaj($noviElement) {
                $this->elementi[] = $noviElement;
                $this->broj++;
        }
```

naziv klase

varijable klase

kostruktor klase

destruktor klase

Vlastita klasa - Red / 5.

```
<? include('Red_1.php');
$red = new Red;
$red->dodaj("Jedan");
$red->dodaj("Dva");
$red->dodaj("Tri");
$red->dodaj("Pet");
brojElemenata($red->broj());
ispisiElemente($red->elementi());
$elem = $red->uzmi();
print "Uzet element: $elem <br>";
brojElemenata($red->broj());
ispisiElemente($red->elementi());

function brojElemenata($broj) {
        print "Ukupno elemenata: $broj<br>";
}

function ispisiElemente($lista) {
        foreach ($lista as $i) {
                print $i . "<br>";
        }
}
```

Primjer40_1 – Vlastita klasa - Red

Vlastita klasa - Stog

```
<? include('Red.php');

class Stog extends Red
{
        function dodaj($noviElement) {
                array_unshift($this->elementi, $noviElement);
                $this->broj++;
        }

        function uzmi() {
                if($this->broj == 0) {
                        return null;
                }
                $odlazi = array_pop($this->elementi);
                $this->broj--;
                return $odlazi;
        }
}
?>
```

uključenje klase

proširenje klase

Vlastita klasa - Stog / 1.

```
<? include('Stog.php');
$stog = new Stog;
$stog->dodaj("Jedan");
$stog->dodaj("Dva");
$stog->dodaj("Tri");
$stog->dodaj("Pet");
brojElemenata($stog->broj());
ispisiElemente($stog->elementi());
$elem = $stog->uzmi();
print "Uzet element: $elem <br>";
brojElemenata($stog->broj());
ispisiElemente($stog->elementi());

function brojElemenata($broj) {
        print "Ukupno elemenata: $broj<br>";
}

function ispisiElemente($lista) {
        foreach ($lista as $i) {
                print $i . "<br>";
        }
}
?>
```

Primjer40_2 – Vlastita klasa - Stog

Rad s datumom i vremenom

Datumski i vremenski podaci imaju značajnu ulogu u mnogim dijelovima Web aplikacija.

`checkdate`	– validacija gregorijanskog datuma
`date`	– formatira lokalno vrijeme/datuma
`getdate`	– daje informaciju u datumu/vremenu
`gettimeofday`	– daje važeće vrijeme
`gmdate`	– formatira GMT lokalno vrijeme/datuma
`gmmktime`	– postavlja UNIX timestamp za GMT datum
`gmstrftime`	– formatira GMT lokalno vrijeme/datum prema lokalnim postavkama
`localtime`	– daje lokalno vrijeme
`microtime`	– daje UNIX timestamp s mikrosekundama
`mktime`	– daje UNIX timestamp za datum
`strftime`	– formatira vrijeme/datuma prema lokalnim postavkama
`strtotime`	– parsira engleski tekstualni opis datuma/vremena u UNIX timestamp
`time`	– daje važeći UNIX timestamp

Rad s datumom i vremenom / 1.

Kodovi za `date` funkciju:

```
d,j     - dan u mjesecu (01-31, 1-31)
D,l     - dan u tjednu (Mon-Sun, Monday-Sunday)
F,M     - mjesec u godini (January-December, Jan-Dec)
g,h     - sat u danu u 12 satnom formatu (1-12, 01-12)
G,H     - sat u danu u 24 satnom formatu (0-23, 00-23)
m,n     - mjesec u godini (01-12, 1-12)
s       - sekunde u minuti (00-59)
t       - ukupan broj dana u mjesecu
U       - ukupan broj sekundi od 01.01.1970.
w       - dan u tjednu (0-6)
y       - godina u 2 brojnom formatu
Y       - godina u 4 brojnom formatu
z       - dan u godini (0-365)
```


Rad s datumom i vremenom / 2.

```php
<?php
echo "Danas je " . date("d.m.Y") . "<br>";
echo "Sada je " . date("H:i:s") . "<br>";
echo "Ovo je " . date("w.") . " dan u tjednu" . "<br>";
echo "Ovo je " . date("z.") . " dan u godini" . "<br>";

$sada = date('d.m.Y H:i:s');
$vrijeme = strtotime($sada) + 60 * 60 * 24;
echo date('F d, Y g:i:s a', $vrijeme);
?>
```

Primjer41 – rad s datumom i vremenom

Vidljivost varijabli

U PHP-u postoje 4 vrste vidljivosti varijabli:

1. Ugrađene superglobalne varijable koje su vidljive bilo gdje u skripti
2. Globalne varijable definirane u skripti vidljive su kroz cijelu skriptu ali ne u funkcijama
3. Varijable definirane u funkcijama su lokalne varijable funkcije
4. Varijable definirane u funkcijama kao globalne odnose se na globalne varijable istog imena (koristi se global $var;)

Superglobalne varijable

$GLOBALS
$_SERVER
$_GET
$_POST
$_FILES
$_COOKIE
$_SESSION
$_REQUEST
$_ENV

Vidljivost varijabli / 1.

Globalne varijable (njihov popis je u **$GLOBALS**):

```
<?php
foreach($GLOBALS as $k => $vr) {
   echo $k . "<br>";
}
?>
```

Uz globalne varijable nalazi se i dio superglobalnih varijabli (samo kraći nazivi):

$_POST
$_GET
$_COOKIE
$_FILES

Primjer26 – Vidljivost varijabli

Vidljivost varijabli / 2.

```
<?php
$a = 1;
echo "a=" . $a . "<br>";
foo();

function foo()
{
   echo "foo: a=" . $a . "<br>";      // varijaba $a nije definirana
   $a = 2;
   echo "foo: a=" . $a . "<br>";
}

echo "a=" . $a . "<br>";
?>
```

Primjer26_1 – Vidljivost varijabli

Vidljivost varijabli / 3.

```
<?php
$a = 1;
echo "a=" . $a . "<br>";
foo();
echo "a=" . $a . "<br>";
function foo()
{
   global $a;
   echo "foo: a=" . $a . "<br>";
   $a = 2;
   echo "foo: a=" . $a . "<br>";
}
?>
```

Primjer27 – Vidljivost varijabli

CGI varijable okoline

Variable Name	Value
DOCUMENT_ROOT	korijenski direktorij na poslužitelju
HTTP_COOKIE	kolačići koje je postavio korisnik
HTTP_HOST	naziv računala
HTTP_REFERER	URL stranice koja je pozvala skriptu
HTTP_USER_AGENT	tip korisnikovog preglednika
HTTPS	"on" ako je skripta zvana preko sigurnog poslužitelja
PATH	putanja na kojoj je izvršen poslužitelj
QUERY_STRING	lista parametara (kod GET poziva)
REMOTE_ADDR	IP adres korisnika
REMOTE_HOST	naziv računala korisnika (ako se može dobiti) ili IP adresa
REMOTE_PORT	port preko kojeg je korisnik povezan na poslužitelj
REMOTE_USER	korisničko ime (ako je stranica zaštićena putem poslužitelja
REQUEST_METHOD	GET ili POST
REQUEST_URI	putanja do zahtjevanog dokumenta ili skripte (relativno do korijenskog direktorija dokumenta)
SCRIPT_FILENAME	puni naziv važećeg sktipte
SCRIPT_NAME	putanja do važeće skripte (relativno do korijenskog direktorija dokumenta)

CGI varijable okoline / 1.

Variable Name	Value
SERVER_ADMIN	email adres webmastera
SERVER_NAME	puno ime poslužitelja
SERVER_PORT	port kojeg prisluškuje poslužitelj
SERVER_SOFTWARE	software koji se koristi na poslužitelju (npr Apache/2.0.52)

WWW poslužitelj automatski postavlja asocijativni niz
$HTTP_SERVER_VARS za svaki CGI – stara verzija php < 4.1.0
$_SERVER za svaki CGI – nova verzija 4.1.0.

Platforma određuje koje su varijable okoline.

CGI varijable okoline / 2.

```
<?php
foreach ($_SERVER as $k => $vr)
        {
        echo "$k=$vr<br>\n";
        }
?>
```

Primjer28 – CGI varijable okoline

Prijenos podataka iz formulara - GET

Za prijenos podataka iz formulara također se koristi varijabla okoline `QUERY_STRING` u koju su upisani podaci svih kontrola iz formulara. Skripta sama obrađuje tu varijablu ali sada je jednostavnije jer se zna format po kojem su upisani podaci. NIJE sigurna metoda slanja podataka jer se podaci šalju kao dio URL-a! U PHP-u postoji asocijativni niz $_GET koji sadrži podatke. Ranije je bila varijabla $HTTP_GET_VARS.

```
<head>
</head>
<body>

<form id="form1" method="get" name="form1" action="primjer29.php">
<p><label for="ime">Ime: </label>
<input name="ime" size="20"><br>
<label for="prezime">Prezime: </label>
<input name="prezime" size="20"><br>
<input id="submit1" name="submit1" type="submit" value="Šalji">
<input id="reset1" name="reset1" type="reset" value="Obriši"> </p>
</form>

</body>
</html>
```


Prijenos podataka iz formulara - GET / 1.

```
<?php
foreach ($_GET as $k => $vr)
        {
        echo "$k=$vr<br>\n";
        }
?>
```

Primjer29 – Prijenos podataka metodom GET

Prijenos podataka iz formulara - POST

Za prijenos podataka iz formulara koristi se standardni ulaz. Skripta čita podatke sa standardnog ulaza i puni varijable (obično tip hash) podacima iz formulara. Sigurnija metoda slanja podataka od GET.

```
<html>
<head>
</head>
<body>

<form id="form1" method="post" name="form1" action="primjer30.php">
<p><label for="ime">Ime: </label>
<input name="ime" size="20" /><br />
<label for="prezime">Prezime: </label>
<input name="prezime" size="20" /><br />
<input id="submit1" name="submit1" type="submit" value="Šalji" />
<input id="reset1" name="reset1" type="reset" value="Obriši" /> </p>
</form>

</body>
</html>
```


Prijenos podataka iz formulara - POST / 1.

WWW poslužitelj provodi određeno pretvaranje nekih znakova:

- **alfanumeričke prenosi bez promjene**
- **praznine pratvara u +**
- **ostale znakove pretvara u oblik `%HH` gdje je `HH` ASCII kod znaka u bazi 16.**

To znači da CGI skripta treba provesti vraćanje u izvorni oblik.
U PHP-u postoji asocijativni niz $_POST koji sadrži podatke. Ranije je bila varijabla $HTTP_POST_VARS.

Prijenos podataka iz formulara - POST / 2.

```
<?php
foreach ($_POST as $k => $vr)
{
        echo "$k=$vr<br>\n";
}
?>
```

Primjer30 – Prijenos podataka metodom POST

Prijenos podataka iz formulara

Prijenos podataka iz formulara različit je za pojedine vrste kontrole kao što su checkbox, radio button i sl. kada se koriste u grupi.

```
<form id="form1" method="post" name="form1" action="primjer30.php">
  <p>Odaberi omiljeni programski jezik: <br />
  <input type="checkbox" name="p_jezik1" value="1" /> C/C++<br />
  <input type="checkbox" name="p_jezik2" value="1" /> Basic<br />
  <input type="checkbox" name="p_jezik3" value="1" checked="checked" /> PHP<br />
  <input type="checkbox" name="p_jezik4" value="1" /> Java<br />
  <input type="checkbox" name="p_jezik5" value="1" />C#<br />
  <input type="submit" value="Šalji" /> <input type="reset" value="Obriši" />
  </p>
</form>
```

Primjer30_1 – Prijenos podataka iz formulara

Prijenos podataka iz formulara

Prijenos podataka iz formulara različit je za pojedine vrste kontrole kao što su checkbox, radio button i sl. kada se koriste u grupi.

```
<form id="form1" method="post" name="form1" action="primjer30.php">
  <p>Odaberi omiljeni programski jezik: <br />
  <input type="checkbox" name="p_jezik" value="1" /> C/C++<br />
  <input type="checkbox" name="p_jezik" value="2" /> Basic<br />
  <input type="checkbox" name="p_jezik" value="3" checked="checked" /> PHP<br />
  <input type="checkbox" name="p_jezik" value="4" /> Java<br />
  <input type="checkbox" name="p_jezik" value="5" />C#<br />
  <input type="submit" value=" Šalji " /> <input type="reset" value=" Obriši " />
  </p>
</form>
```

Primjer30_2 – Prijenos podataka iz formulara

Prijenos podataka iz formulara

Prijenos podataka iz formulara različit je za pojedine vrste kontrole kao što su checkbox, radio button i sl. kada se koriste u grupi.

```
<form id="form1" method="post" name="form1" action="primjer30_2_1.php">
 <p>Odaberi omiljeni programski jezik: <br />
 <input type="checkbox" name="p_jezik[]" value="1" /> C/C++<br />
 <input type="checkbox" name="p_jezik[]" value="2" /> Basic<br />
 <input type="checkbox" name="p_jezik[]" value="3" checked="checked" /> PHP<br />
 <input type="checkbox" name="p_jezik[]" value="4" /> Java<br />
 <input type="checkbox" name="p_jezik[]" value="5" />C#<br />
 <input type="submit" value=" Šalji " /> <input type="reset" value=" Obriši " />
 </p>
</form>
```

Primjer30_2_1 – Prijenos podataka iz formulara

Prijenos podataka iz formulara

```
<?php
 foreach ($_POST as $k => $vr) {
    if (is_array($vr)) {
        foreach ($vr as $i => $j) {
            echo "$i=$j<br>\n";
        }
    }
    else
        echo "$k=$vr<br>\n";
}?>
```

Primjer30_2_1 – Prijenos podataka metodom POST

Prijenos podataka iz formulara

Prijenos podataka iz formulara različit je za pojedine vrste kontrole kao što su checkbox, radio button i sl. kada se koriste u grupi.

```
<form id="form1" method="post" name="form1" action="primjer30.php">
  <p>Odaberi omiljeni programski jezik: <br />
  <input type="radio" name="p_jezik1" value="1" /> C/C++<br />
  <input type="radio" name="p_jezik2" value="1" /> Basic<br />
  <input type="radio" name="p_jezik3" value="1" checked="checked" /> PHP<br />
  <input type="radio" name="p_jezik4" value="1" /> Java<br />
  <input type="radio" name="p_jezik5" value="1" />C#<br />
  <input type="submit" value="Šalji" /> <input type="reset" value="Obriši" />
  </p>
</form>
```

Primjer30_3 – Prijenos podataka iz formulara

Prijenos podataka iz formulara

Prijenos podataka iz formulara različit je za pojedine vrste kontrole kao što su checkbox, radio button i sl. kada se koriste u grupi.

```
<form id="form1" method="post" name="form1" action="primjer30.php">
  <p>Odaberi omiljeni programski jezik: <br />
  <input type="radio" name="p_jezik" value="1" /> C/C++<br />
  <input type="radio" name="p_jezik" value="2" /> Basic<br />
  <input type="radio" name="p_jezik" value="3" checked="checked" /> PHP<br />
  <input type="radio" name="p_jezik" value="3" /> Java<br />
  <input type="radio" name="p_jezik" value="5" />C#<br />
  <input type="submit" value=" Šalji " /> <input type="reset" value=" Obriši " />
  </p>
</form>>
```

Primjer30_4 – Prijenos podataka iz formulara

Cookie

Programiranje na strani klijenta i na strani poslužitelja moraju imati više dodirnih točaka-sučelja putem kojih se ostvaruje njihova komunikacija.
Prije svega to su standardni ulaz i standardni izlaz na strani poslužitelja kojima se primaju podaci od klijenta i kojima se šalju podaci klijentu.
Druga važna točka su varijable okoline koje služe strani poslužitelja za prikupljanje podataka o klientu i njegovom radnom okruženju.
Personalizacija se obavlja na strani klijenta, a identifikator korisnika se zapisuje kao vrijednost određenog cookie-a.
Ostaje pitanje kako poslužiteljska strana čita i zapisuje cookie?

Upisivanje:

```
int setcookie (string name [, string value [, int expire [,
string path [, string domain [, int secure]]]]])
```

Danas se cookie najčešće koristi za spremanje korisničkog imena kod zadnjeg uspješnog prijavljivanja. (npr. Moodle, GMail i sl.). Google ga koristi za personalizaciju (jezik i sl.)

Postavljanje Cookie-a

```
<html>
<head>
<title>PHP - Primjer 37</title>
</head>
<body>
<?php
$naziv = "WebDiP";
$id = "dkermek";
$vrijedi_do = time() + 20; // vrijedi 200 sek.

setcookie($naziv, $id, $vrijedi_do);
print "<b>Cookie:</b> $naziv <b>vrijedi do:</b> $vrijedi_do.\n";
?>
```

Primjer37 – Postavljanje Cookiea

Postavljanje Cookie-a / 1.

```
<?php
$naziv = "PzaWeb";
$id = "dkermek";
$vrijedi_do = time() + 60*60*24*7;
setcookie($naziv, $id, $vrijedi_do);
?>
<html>
<head>
<title>PHP - Primjer 37_1</title>
</head>
<body>
<?php
print "<b>Cookie:</b> $naziv <b>vrijedi do:</b> $vrijedi_do.\n";
?>
</body>
</html>
```

Primjer37_1 – Postavljanje Cookiea - u redu

Čitanje Cookie-a

Varijabla $_COOKIE sadrže cookie. Ranije je bila varijabla $HTTP_COOKIE_VARS

```
<?php
$naziv = "WebDiP";
$id = $_COOKIE[$naziv];
print "<b>Cookie:</b> $naziv=$id\n";
?>
```

Primjer38 – Čitanje Cookiea

Korištena i dodatna literatura

- http://www.php.net/
- http://www.php.org/
- http://perl.about.com/library/phpCR/bl_index.htm?PM=ss14_perl
- http://linuxdocs.org/HOWTOs/PHP-HOWTO.html
- http://www.php.net/manual/en/install.apache2.php
- http://www.thesitewizard.com/archive/phpvscgi.shtml
- http://www.thesitewizard.com/archive/feedbackphp.shtml
- http://php.resourceindex.com/Documentation/
- http://phpmyadmin.sourceforge.net/
- http://www.oreilly.com/catalog/phpckbk/chapter/ch08.pdf
- http://www.oreilly.com/catalog/progphp/chapter/ch05.html
- http://www.devshed.com/Server_Side/PHP