



Prirodoslovno-matematički fakultet  
Matematički odsjek  
Sveučilište u Zagrebu

# RAČUNARSKI PRAKTIKUM II

Predavanje 04 - Uvod u PHP

10. ožujka 2020.

Sastavio: Zvonimir Bujanović



## PHP

- Interpretirani skriptni jezik namijenjen gotovo isključivo serverskoj strani web-programiranja.
- Prva verzija 1994. kao "Personal Home Page".
- Aktualna verzija 7.4 kao "PHP: Hypertext Preprocessor".
- Open source.
- Višeplatformski; podržan unutar mnogo različitih web-servera.
- Poznati web-ovi koji su implementirani u PHP-u: Yahoo, Wikipedia, Facebook, WordPress.

Ostali jezici na serverskoj strani:

- Ruby on Rails [Github, Indiegogo, Hulu, prije Groupon, Twitter]
- Python – Django [Pinterest, Instagram, Disqus, Bitbucket]
- JavaScript (Node.js) [Walmart, PayPal, LinkedIn, Groupon]
- ASP.NET [StackOverflow]
- Java Server Pages (JSP)

## Knjige:

- L. Ullman – *PHP and MySQL for Dynamic Web Sites*, 2011.
- L. Welling, L. Thomson – *PHP and MySQL Web Development*, 2008.
- D. Sklar, A. Trachtenberg – *PHP Cookbook*, 2014.
- J. Lockhart – *Modern PHP*, 2015.

## Web-resursi:

- PHP Manual
- w3schools - PHP Tutorial
- Tutorialspoint PHP
- Phptpoint Tutorial
- The new Boston - 200 video tutoriala

index.php:

```
1 <!DOCTYPE html>
2 <html lang="hr">
3 <head>...</head>
4 <body>
5     <?php
6         echo "<p>Hello world!</p>";
7     ?>
8 </body>
```

- 1 Kljent se spaja na web-server tražeći `index.php`.
- 2 Web-server detektira ekstenziju, te pokreće PHP-interpreter s ulaznom *source* datotekom `index.php`.
- 3 PHP-interpreter izvršava naredbe unutar `<?php` i `?>` tagova i tako generira tekst-datoteku.
- 4 Dobivena datoteka se šalje klijentu.

Sintaksa vrlo slična C-u:

- case-sensitive (više-manje)
- komentari `// ...` i `/*...*/`
- naredbe završavaju sa `;`
- blokovi naredbi unutar `{...}`
- moguće uključiti drugu datoteku sa `include 'druga.php';`

Variable

- Ime varijable mora počinjati sa `$`, na primjer `$x = 5;`
- Varijable se ne deklariraju.
- Varijable mogu mijenjati tip prilikom izvršavanja.
- Postoji mnogo predefiniраних varijabli (o njima kasnije).
- Može se testirati da li postoji sa `isset($var)`.
- Može se uništiti sa `unset($var)`.

# Tipovi podataka

Postoji 8 primitivnih tipova podataka:

- `boolean`, `integer`, `float` (dvostruka preciznost), `string`;
- `array`, `object`;
- `resource`, `null`.

- `boolean`

```
1 $foo = True; $bar = FALSE;
```

Drugi tipovi se mogu automatski konvertirati u `boolean`:

```
1 $str = "hello";  
2 if( $str ) echo "String hello je true.";
```

Moguć je i eksplicitni cast sa `(boolean)$str`, kao u C-u.

- **integer**
  - Uobičajeni cjelobrojni tip s predznakom.
  - Raspon ovisi o platformi (32bit vs 64bit).
  - U slučaju overflowa, automatska konverzija u float!
  - Tek u PHP7 je uvedeno cjelobrojno dijeljenje; vidi `round()`, `floor()`, `intdiv()`.
  - PHP7: **spaceship operator** (radi i za float).
  - Konverzija u integer sa `(int)$var` ili `(integer)$var`

```
1 $million = 1000000; $big = 5000000000000000 * $million;
2 var_dump( $million );      // int(1000000)
3 var_dump( $big );          // float(5.0E+19)
4 var_dump( 25/7 );          // float(3.5714285714286)
5 var_dump( (int)(25/7) );   // int(3)
6
7 // Spaceship!
8 $a = 3; $b = 5; $x = ( $a <=> $b ); // $x = 1
```

## Zadatak 1

Napišite PHP skriptu koja generira HTML tablicu množenja brojeva od 1 do 10.

*	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100



- string

- Niz znakova, svaki zauzima 1 byte (PHP nativno ne podržava Unicode; vidi [mbstring](#) i [popis funkcija](#)!)
- String literal može biti unutar `"..."` i `'...'`. **Nije isto!**
- Samo unutar `"..."` se evaluiraju vrijednosti varijabli.
- Samo unutar `"..."` rade *escape-sekvence*: `\n`, `\"`, `\\`, `\$`, ...

```
1 $str = 'world';  
2 echo "Hello, $str!\n";    // Hello, world!  
3 echo 'Hello, $str!\n';    // Hello, $str!\n
```

- Konkatenacija sa `.`: na primjer, `$x = 'Hello ' . 'world';`
- Pristup pojedinim znakovima pomoću `[]`:  
na primjer, `$str[2] = 'x';`
- Postoje brojne funkcije za rad sa stringovima, slične kao u C-u:  
[strlen](#), [substr](#), [strcmp](#), [sprintf](#), [sscanf](#), ...

# Tipovi podataka

- array

- Polja su zapravo mape, tj. preslikavanja ključ  $\mapsto$  vrijednost.
- Ključ je integer ili string, vrijednost je proizvoljna.
- Postoje brojne funkcije za rad s poljima, npr. `sort`.
- Osim ključne riječi `array` za deklaraciju se može koristiti i `[]`.

```
1 $polje = array( 6, 3, 7, 9 );
2 for( $i = 0; $i < 4; $i++ )
3     echo $polje[$i] . ' '; // ispiše: 6 3 7 9
4
5 $polje = array( 'bla' => 'foo', 2 => 'bar', 19 => 'oo' );
6 $polje['abc'] = 9;
7 unset( $polje[19] );
8
9 foreach( $polje as $key => $value )
10     echo $key . ' => ' . $value . ', ';
11 // ispiše: bla => foo, 2 => bar, abc => 9
12
13 $polje = [6, 3, 7, 9]; // OK
14 $polje = ['bla' => 'foo', 2 => 'bar', 19 => 'oo']; // OK
```

Napišite PHP skriptu koja:

- 1 Slučajno generira niz od 10 stringova – nizova znakova od po 5 slova.
- 2 Ispisuje generirane stringove.
- 3 Sortira taj niz stringova, te ga ispisuje nakon sortiranja.

Linkovi: `rand()`, `chr()`, `strcmp()`.

Postoji i funkcija `sort()`. Nemojte ju koristiti u ovom zadatku.

- `null`

- Jedina vrijednost tipa `null` je `NULL` (nije case-sensitive :)).
- Varijabla je tipa `null` ako joj je pridružena konstanta `NULL`;
- ili ako joj nije dosad bila pridružena nikakva vrijednost;
- ili ako je uništena sa `unset()`.

# Operatori, kontrola toka, petlje

- Operatori kao u C-u; iznimka `==` i `===`, te `!=` i `!==`.
- `a==b` radi konverziju ako `a` i `b` nisu istog tipa. **Izbjegavati!**
- `a===b` vraća `false` ako `a` i `b` nisu istog tipa. **Koristiti!**
- Uz `+=`, `-=` i slične, postoji i `.=` za konkatenciju.
- `if`, `switch`, `while`, `do...while`, `for`, `break`, `continue` identični kao u C-u. Moguće su kombinacije PHP i ne-PHP koda:

```
1  for( $i = 0; $i < 5; ++$i ) { ?>
2      <p>Hello, <?php echo $i; ?>!</p>
3      <?php
4  }
```

- **foreach** – iteracija po svim elementima polja.

```
1  $polje = array( 6, 3, 7, 9 );
2  foreach( $polje as &$val ) // uočiti & - referenca!
3      $val = 2*$val; // bez & se ne smije mijenjati $val
4  print_r( $polje ); // 12, 6, 14, 18
```

## PHP7: *null coalescing operator*

Često ćemo trebati pisati kod poput:

```
1 if( isset( $_SESSION['user'] ) )
2     $user = $_SESSION['user'];
3 elseif( isset( $_POST['user'] ) )
4     $user = $_POST['user'];
5 else
6     $user = 'guest';
```

PHP7 uvodi *null coalescing operator* ??

- Vraća vrijednost prve varijable u slijedu koja postoji (nije `null`).

```
1 $user = $_SESSION['user'] ?? $_POST['user'] ?? 'guest';
```

- Imena funkcija nisu case-sensitive?!
- Argumenti se mogu slati po vrijednosti ili po referenci.
- Nekoliko zadnjih argumenata može imati defaultnu vrijednost (kao u C++).
- Overloadanje funkcija nije podržano.

```
1 function foo( $x, $y = "zz" ) { $x .= $y; return $x; }
2
3 function bar( &$amp;x, $y = "zz" ) { $x .= $y; return $x; }
4
5 $x = 'Hello, '; $y = 'world!'; $z = foo( $x, $y );
6 echo $z; // 'Hello, world!'
7 echo $x; // 'Hello, '
8
9 $z = bar( $x );
10 echo $z; // 'Hello, zz!'
11 echo $x; // 'Hello, zz!'
```

# Funkcije

- Podržane su anonimne funkcije (kao i u JavaScriptu).
- Globalne varijable se moraju specijalno deklarirati.
- Sa `static` se mogu definirati statičke varijable unutar funkcija.

```
1 $x = 1; $y = 5;
2
3 function foo() {
4     global $x;
5
6     var_dump( $x ); // int(1)
7     var_dump( $y ); // NULL
8 }
9
10 foo();
11
12 $f = function( $a ) { return $a*$a; };
13 $z = $f( 5 ); // uoči $ ispred f
14 echo $z;      // 25
```



## Zadatak 3

Modificirajte rješenje **Zadatka 2** tako da postoji funkcija `my_sort()` koja prima polje, te ga sortira.

Možete koristite `count()` za određivanje broja elemenata polja.

- Koncept je sličan kao u C++/Java, razlike u sintaksi.
- Članovi klasa su svojstva i metode, mogu biti `private`, `protected` i `public`.
- Po defaultu su metode `public`.
- Postoji `$this` "pointer" unutar metoda. Nije implicitan!
- Konstruktor je `__construct`, a destruktor je `__destruct` (pripadaju tzv. **magičnim metodama**).
- Objekti klase koja ima metodu `__toString` mogu se konvertirati u `string` (i stoga ispisivati sa `echo`).
- Metode i svojstva mogu biti `static`.
- Novi objekti se stvaraju sa `new`.
- Funkcija može promijeniti svojstva objekta poslanog kao argument.

# Objektno orijentirano programiranje

```
1 class Test
2 {
3     private $data;
4     public $msg = 'hello';
5     static $info = 'info';
6
7     function __construct( $x ) { $this->data = $x; }
8     function __destruct() { echo 'Destruktor!'; }
9     function getdata() { return $this->data; }
10 };
11
12 function foo( $x ) { $x->msg = 'ooo'; }
13
14 $t = new Test( 5 );
15 echo $t->msg;           // 'hello'
16 echo $t->getdata();     // 5
17 echo Test::$info;      // 'info'
18
19 foo( $t ); echo $t->msg; // 'ooo'
20 // 'Destruktor!'
```

- Nasljeđivanje pomoću ključne riječi **extends**.
- Sve **public** i **protected** funkcije su po *defaultu* virtualne. Ako to ne želimo  $\rightsquigarrow$  deklarirati funkciju u baznoj klasi sa **final**.
- Nije moguće višestruko nasljeđivanje.
- Pristup roditelju pomoću **parent::svojstvo** ili **parent::metoda()**.
- Konstruktor i destruktor ne pozivaju automatski roditeljeve metode.
- Moguće definirati metodu kao **abstract**. Tada i cijela klasa treba biti označena sa **abstract**.

## Objektno orijentirano programiranje - Nasljeđivanje

```
1 abstract class Lik {
2     public $ime = 'Lik';
3     function __construct() { echo 'Lik::__con'; }
4     abstract protected function opseg();
5 };
6
7 class Kvadrat extends Lik {
8     private $x;
9     function __construct( $x ) {
10         parent::__construct();
11         echo 'Kvadrat::__con';
12         $this->x = $x;
13     }
14
15     function opseg() { return 4*$this->x; }
16 }
17
18 $k = new Kvadrat( 5 ); // 'Lik::__con' 'Kvadrat::__con'
19 echo $k->opseg();      // 20
20 echo $k->ime;          // 'Lik'
```

# Objektno orijentirano programiranje - Interface

```
1 interface iIzmjeriv
2 {
3     function opseg();
4 };
5
6 // može implementirati i više interfacea, odvojiti zarezom
7 class Kvadrat implements iIzmjeriv
8 {
9     private $x;
10    function __construct( $x ) {
11        echo 'Kvadrat::__con';
12        $this->x = $x;
13    }
14
15    function opseg() { return 4*$this->x; }
16 }
17
18 $k = new Kvadrat( 5 ); // 'Kvadrat::__con'
19 echo $k->opseg();      // 20
```

Napravite hijerarhiju klasa `Vozilo`, `Auto`, `Tramvaj`.

- Svako vozilo ima ime koje dobije u konstruktoru.
- Svako vozilo ima smjer u kojem je okrenuto (N, S, W, E).
- Svako vozilo ima (x, y) koordinate u kojima se trenutno nalazi.
- Funkcija `gdjeSam()` ispisuje ime vozila i trenutne koordinate.
- Sva vozila implementiraju sučelje `IUpravljiv`. U tom sučelju se nalaze funkcije:
  - `idiRavno(x)` – vozi x kilometara u smjeru u kojem je okrenuto
  - `skreniDesno()`, `skreniLijevo()`
- `Auto` troši 10 litara benzina po kilometru, te ima funkciju `potrosenBenzin()` koja vraća ukupnu količinu do sada potrošenog benzina.
- `Tramvaj` ima konstruktor koji prima redni broj linije, te funkciju `linija()` koja vraća taj broj.

Napravite polje s 10 različitih vozila, te pozovite par funkcija iz sučelja i klasa na njima.