



# Programiranje u PHP-u

Predavač:  
Dalibor Bužić

1

1

## Programiranje u PHP



- Koncept:
  - Predavanja
  - Laboratorijske vježbe
  - Projekt
- Literatura:
  - Kevin Tatroe, Peter MacIntyre, Rasmus Lerdorf: "Programming PHP" 2013.
  - <http://www.php.net/manual/en/>
  - Slajdovi
- Obaveze na predavanjima i vježbama
- Polaganje ispita = projekt

2

2

## Predviđene teme



- Uvod u PHP
- Osnove jezika
- Funkcije
- Stringovi
- Nizovi
- Objektno orijentirano programiranje u PHP-u
- Rad s formama
- Rad s bazama podataka
- Grafika u PHP-u
- PDF ekstenzije
- XML i PHP
- Integracija s drugim tehnologijama

3

3

## Skriptiranje

- Nedostatak HTML-a?
- Skripte – ASP,.NET PHP, Perl, JavaScript
- Skriptiranje se može odvijati na strani klijenta i na strani poslužitelja
- Tko je klijent?

4

4

2

## Poslužiteljsko skriptiranje

- Poslužiteljsko (serversko) skriptiranje stvara HTML stranicu temeljem zahtjeva korisnika (npr. prikaz svih proizvoda u kategoriji "Autogume 195×65 R15") i to tako da se podaci na stranici stvaraju dinamički na poslužitelju PRIJE nego se stranica pošalje web pregledniku.
- Korisnik vidi gotovu statičnu web stranicu
- Izvorne naredbe poslužiteljskih skripti nisu vidljive korisniku
- Serverski skriptni jezici su PHP, ASP.NET, ColdFusion, JSP, Perl i drugi.

5

5

## Klijentsko skriptiranje

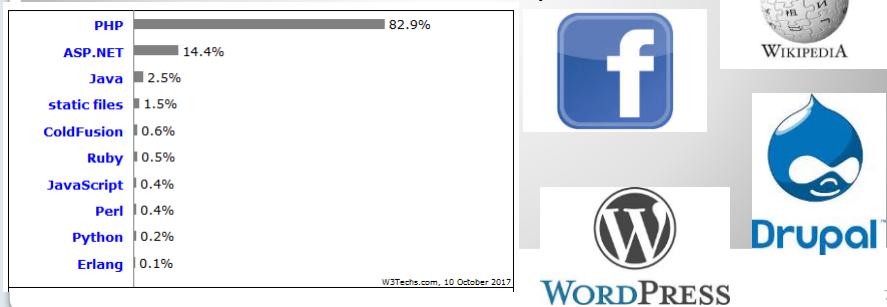
- Klijentsko skriptiranje omogućuje interaktivnost web stranice NAKON što se pošalje web pregledniku.
- Sva obrada se vrši na korisnikovom računalu.
- Primjeri korištenja klijentskog skriptiranja su računanje iznosa u košarici kupca, provjera podataka unesenih u obrazac (radi ubrzavanja procedure izbjegava se slanje podataka na poslužitelj koji također može obavljati takvu provjeru) te izmjena izgleda web stranice.
- Izvorne naredbe klijentskih skripti vidljive su svakom posjetitelju ako pregledava *source* stranice.
- Najpopularniji klijentski skriptni jezik je JavaScript.

6

6

## Programski jezik PHP

- PHP je open-source server-side skriptni programski jezik koji služi za izradu dinamičkih web stranica.
- Izrazito je rasprostranjen, prema podacima W3Techs iz listopada 2017, na preko 80% websiteova koriste se PHP skripte.



7

## Programski jezik PHP

- Prvu verziju napravio je Rasmus Lerdorf 1995. godine pod nazivom Personal Home Page Tools (PHP Tools).
- Iduće godine predstavljena je verzija PHP/FI (FI dolazi od Form Interpreter) koja omogućuje interakciju s bazom podataka.
- Verziju 3 s novim parserom Lerdorf razvija s Zeevom Suraskijem i Andijem Gutmansom 1998. godine.
- Četvrta verzija, opet s novim parserom imena Zend, predstavljena je 2000. godine.
- Posljednja verzija je 7.1.

8

8

## Programski jezik PHP

- PHP se može pokrenuti na svim važnijim operacijskim sustavima – Windows, Unix (Linux, Ubuntu, Debian, Solaris, FreeBSD) i Mac OS X.
- Također se može koristiti na svim vodećim web poslužiteljima kao što su Apache Microsoft IIS i Netscape/iPlanet.
- PHP podržava rad sa najraširenijim SUBP-ovima: MySQL, Oracle, MS SQL Server, Sybase, DB2, PostgreSQL, SQLite, MongoDB...
- Osim HTML dokumenta kao izlaza (rezultata), PHP može generirati i druge vrste dokumenata: PDF, JPEG, GIF, PNG, Flash...

9

9

## Osnove PHP-a

- PHP je dijelom naslijedio sintaksu programskog jezika C, a u sintaksi je vidljiv i utjecaj Perl-a.
- PHP naredbe se mogu ugraditi u HTML kod, i to tako da se PHP kod odvoji od HTML-a pomoću <? i ?> ili <php i ?> tagova

```
<html>
    <head>
        <meta charset="UTF-8">
        <title>Prva PHP skripta</title>
    </head>
    <body>
        <?php
            for ($i=0; $i<5;$i++)
                echo("Hello VSITE! <br>");
        ?>
    </body>
</html>
```

10

10

## Osnove PHP-a

- Naredbe se u PHP-u međusobno odvajaju točka-zarezom (;). Točka-zarez se može izostaviti neposredno prije zatvarajućeg PHP taga:

```
<?php  
    echo("Hello VSITE! <br>")  
?>
```

- ne smije se izostaviti ispred zatvorene vitičaste zagrade koja oblikuje blok naredbi (primjerice kod petlji ili if naredbe):

```
<?php  
for ($i=0; $i<5;$i++) {  
echo("Hello VSITE! <br>");  
echo("PHP je broj jedan! <br>"); } //točka-zarez je obavezna  
?>
```

11

11

## Osnove PHP-a

- Imena korisnički definiranih klasa i funkcija te ključnih riječi nisu slovčano osjetljiva, što znači da će se naredbe

```
ECHO("Hello!")  
echo("Hello!")  
eChO("Hello!")
```

izvršiti jednak, dakle, naredbe su ekvivalentne.

- ALI, imena varijabli JESU slovčano osjetljiva pa su \$placa, \$PLACA i \$pLaCa tri različite variable!
- Općenito, razmak ne utječe na izvođenje PHP programa. Jednu naredbu moguće je 'razbiti' na više redova.
- U jednom redu moguće je napisati više naredbi, naravno, potrebno ih je odvojiti točka-zarezom.

12

12

## Osnove PHP-a

- Komentare u kodu potrebno je uključivati kod naredbi za koje nije očito što rade, a služe kao pomoć samom sebi, ali i drugim programerima i u velikoj mjeri olakšavaju održavanje koda.

- Kod se može komentirati na tri načina:

```
<?php  
    for ($i=0; $i<5;$i++) { #ovo je komentar  
        echo("Hello VSITE! <br>");  
        echo("PHP je broj jedan! <br>"); } //također komentar  
/* ali i ovo je komentar  
   a proteže se na više redova */  
  
?>
```

- Dakle, komentar može započeti znakom # i on vrijedi do kraja tekućeg reda ili kraja sekcije PHP koda (ovisno što dolazi prije).
- Isto vrijedi i ako komentar započne dvjema kosim crtama //.

13

13

## Osnove PHP-a - identifikatori

- Identifikatori su imena varijabli, funkcija, klase i konstanti. Identifikatori se mogu sastojati samo od slova (velika ili mala), znaka podvlake i brojeva. Identifikator ne smije početi brojem.

- Imena varijabli moraju početi znakom \$. Ispravno ili ne???:

```
$placa  
$prosjecna placa  
$Prezime  
$!  
$1broj  
$prirodni_broj  
$brojac2
```

- Zahvaljujući razlikovanju malih i velikih slova sve niže navedene varijable su različite:

```
$datum_zaposlenja  
$Datum_Zaposlenja  
$DATUM_ZAPOSLENJA  
$dATUM_zAPOSLENJA
```

14

14

## Osnove PHP-a - identifikatori

- Imena funkcija nisu slovčano osjetljiva, pa se sve niže navedena imena odnose na istu funkciju:

PrikaziDatum() prikazidatum()  
pRIKAZIdATUM() Prikazidatum()  
PRIKAZIDATUM()

- Konstante su slovčano osjetljive, a mogu biti jednog od sljedećih tipova: Boolean, integer, double ili string. Konstante se deklariraju pomoću define() funkcije:

```
define('PI', 3.14);  
echo PI;
```

15

15

## Tipovi podataka

- PHP ima osam podatkovnih tipova.
- Četiri tipa su skalari (imaju jednu vrijednost):
  - cijeli brojevi (integer),
  - realni brojevi (brojevi s pomičnim zarezom),
  - stringovi i
  - Boolean tip podatka.
- Dva tipa su složena: nizovi i objekti, a dva tipa su posebna: resource i NULL.
- Cjelobrojni tip uobičajeno (ali ne i na svim sustavima!) sadržava neku od vrijednosti između -2.147.483.648 i 2.147.483.647.
- Realni brojevi se prikazuju u dva formata: uobičajeni (npr. -345.123, 678000.01) i znanstveni prikaz (npr. 32.4E-7 što je  $32.4 \times 10^{-7}$  ili 0.00000324).

16

16

## Tipovi podataka

- Stringovi se navode unutar jednostrukih ili dvostrukih navodnika.
- Razlika postoji, kao što pokazuje sljedeći primjer:

```
$lik="Crvenkapica";
echo "$lik sreće vuka<br>";
echo '$lik sreće vuka<br>';
```

**Crvenkapica sreće vuka  
\$lik sreće vuka**

17

17

## Boolean vrijednosti

- Boolean vrijednosti su dvije: true ili false.
- U PHP-u false je:
  - ključna riječ false
  - cijeli broj 0
  - decimalni broj 0.0
  - prazan string "" ali i "0"
  - niz s nula elemenata
  - objekt bez vrijednosti ili funkcija
  - vrijednost NULL
- Ono što nije false, je true!

18

18

## Varijable

- imena varijabli moraju imati prefiks \$ (znak dolara).
- Varijabla može sadržavati bilo koji tip podatka.
- Varijabla može u nekom trenutku sadržavati tekstualni podatak, a već u sljedećem cjelobrojnu vrijednost!

```
$svemoze="Crvenkapica";  
$svemoze=2001;  
$svemoze=array("vuk", "lovac", 48);
```

19

19

## Opseg varijabli

- Varijable mogu imati lokalni ili globalni opseg, te mogu biti statične.
- Varijabla deklarirana u funkciji ima lokalni opseg, dakle vidljiva je jedino u toj funkciji.
- Nebitno je da li je varijabla deklarirana unutar bloka (npr. petlje).

20

20

## Opseg varijabli

- Varijabla deklarirana izvan funkcije (globalna varijabla) nije vidljiva nijednoj funkciji, osim ako se ne postupi na sljedeći način: deklarirati varijablu pomoću ključne riječi global, te to napraviti unutar funkcije!

```
function promijeni()
{
    global $x;
    $x++;
}
$x=5;
promijeni();
echo $x; //ispisuje [ ]
```

21

21

## Opseg varijabli

- Statične varijable su vidljive samo unutar funkcije u kojoj su stvorene, no životni vijek im je produžen – one ostaju u memoriji i nakon što izvođenje napusti funkciju.

```
function promijeni()
{
    static $x=0;
    $x++;
    echo "statični x je sad {$x} <br>"; //ispisuje [ ]
}
$x=5;
promijeni();
promijeni();
echo "globalni x je sad {$x} <br>"; //ispisuje [ ]
```

- Parametri funkcije lokalne varijable, vidljive su samo unutar te funkcije.

22

22

## Operatori

- Aritmetički:

zbrajanje	$\$x + \$y$
oduzimanje	$\$x - \$y$
množenje	$\$x * \$y$
dijeljenje	$\$x / \$y$
Ostatak pri dijeljenju	$\$x \% \$y$
Unarni plus	$+(\$x * \$y)$
Unarni minus	$- (\$x * \$y)$
potenciranje	$\$x ** \$y$

23

23

## Operatori

- dijeljenja

```
$y=37;  
$a=4;  
$b=$y/$a;  
echo $b . '<br>'; // ispisuje [redacted]  
$y=30.6;  
$a=4;  
$b=$y%$a;  
echo $b . '<br>'; // ispisuje [redacted]
```

24

24

## Operatori

- Operator spajanja stringova .

```
$a=7;  
$s= 'Vuk i '. $a . ' kozlića';  
echo $s;
```

25

25

## Operatori ++ i --

- Operatori uvećanja i umanjenja ++ --

++\$x znači  $\$x = \$x + 1$   
\$x++ znači  $\$x = \$x + 1$   
--\$x znači  $\$x = \$x - 1$   
\$x-- znači  $\$x = \$x - 1$

- kad su ovi operatori sastavni dijelovi izraza, bitno je da li se nalaze ispred ili iza varijable:
  - ako je operator ispred varijable, računa se s uvećanom (umanjenom) vrijednošću
  - ako je operator iza varijable, varijabla se uvećava (umanjuje) NAKON izračunavanja izraza

26

26

## Operatori ++ i --

```
$a=4;  
$b= ++$a + 2;  
$a=4;  
$b= $a++ +2;
```

Koliko je \$b nakon druge, a koliko nakon četvrte naredbe?

27

27

## Operatori ++ i --

- ovi operatori se mogu primijeniti i na string varijable!

```
$tekst="abc";  
$tekst++;  
echo $tekst; //ispisuje abcabc
```

28

28

## Operatori usporedbe

- == jednako
- === identično
- != ili <> različito
- !== nije identično
- > veće od
- >= veće ili jednako
- < manje od
- <= manje ili jednako
- Rezultati operacija su true ili false

29

29

## Operatori usporedbe

- === moraju imati jednaku vrijednost i biti istog tipa

```
$a=5;  
$b=5.0;
```

```
$a===$b;  
$a===$b;
```



30

30

## Kombinirani operator usporedbe

`<=>` 'spaceship' operator

```
echo 1 <=> 1;  
echo 1 <=> 2;  
echo 2 <=> 1;
```

31

31

## Bitovni operatori

- promatraju operative kao skupove bitova
- `&` bitovni I
- `|` bitovni ILI
- `^` bitovni ekskluzivni ILI
- `~` bitovni NE

```
$a= 10 & 13;  
echo "$a <br>"; // ispisuje  
$a= 10 | 13;  
echo "$a <br>"; // ispisuje  
$a= 10 ^ 13;  
echo "$a <br>"; // ispisuje  
$a= ~10 ;  
echo "$a <br>"; // ispisuje
```

32

32

## Logički operatori

Slovčani	Znakovni	značenje
and	&&	I
or		ILI
Xor		Ekskluzivni ILI
	!	NE

- rezultati su true odnosno false
- and i or su kratkospojni operatori
- slovčana i znakovna varijanta operatara I nemaju isti prioritet!
- isto vrijedi i za operator ILI

33

33

## Operatori pretvorbe tipova

operator	Pretvara u tip
(int), (integer)	integer
(bool), (boolean)	boolean
(float), (double), (real)	floating point
(string)	string
(array)	array
(object)	object
(unset)	NULL

```
$a=5.9;  
$a= (int) $a;  
echo $a; // ispisuje
```

34

34

## Operatori pridruživanja

- = operator pridruživanja

```
$a=5;  
$b=2;  
$c=$a=$b; // razumljivo je [ ]  
echo $c; //ispisuje [ ]
```

35

35

## Operatori pridruživanja

- operatori skraćenog pridruživanja

Operator	primjer	značenje
+=	\$x+=2;	\$x=\$x+2;
-=	\$x-=2;	\$x=\$x-2;
*=	\$x*=2;	\$x=\$x*2;
/=	\$x/=2;	\$x=\$x/2;
%=	\$x%=2;	\$x=\$x%2;
&=	\$x&=2;	\$x=\$x&2; //bitovni I
=	\$x =2;	\$x=\$x 2; //bitovni ILI
^=	\$x^=2;	\$x=\$x^2; //bitovni ekskl. ILI
.=	\$x.="2";	\$x=\$x . "2";

36

36

## Uvjetni operator

- ? : uvjetni (ternarni operator)

```
uvjet? true_naredba : false_naredba;
```

```
2>1? $a='da':$a='ne';
echo $a; //ispisuje 
```

37



## P2: Kontrola toka programa

Kolegij: Programiranje u PHP  
Predavač: Dalibor Bužić

## Kontrola toka programa

- ▶ Naredbe se uobičajeno izvršavaju slijedno
- ▶ Često je slijedno izvršavanje potrebno narušiti pomoću
  - selekcijskih naredbi (grananja, naredbe odluke)
    - if
    - if ... else
    - if ... elseif ... else
    - switch
  - naredbi ponavljanja (programske petlje)
    - while
    - do ... while
    - for
    - foreach

## Naredba If

- ▶ izvršava naredbu ako je uvjet true  
`if (uvjet) naredba;`
- ▶ za više naredbi koje treba izvršiti ako je uvjet ispunjen, koristi se blok naredbi koji se omeđuje s { na početku te s } na kraju bloka

```
if (uvjet) {
    naredba_1;
    naredba_2;

    ...
    naredba_n;
}
```

3

3

## Naredba If ... else

- ▶ izvršava jednu naredbu ako je uvjet true, a drugu ako je uvjet false

```
if (uvjet)
    naredba1;
else
    naredba2;
```

- ▶ za više naredbi koje treba izvršiti ako je uvjet ispunjen (ili neispunjen), koristi se blok naredbi koji se omeđuje s { na početku te s } na kraju bloka

```
if (uvjet) {
    naredba_1;
    naredba_2;
    ...
    naredba_n;
}
else {
    naredba_A1;
    naredba_A2;
    ...
    naredba_An;
}
```

4

4

## Naredba If ... else

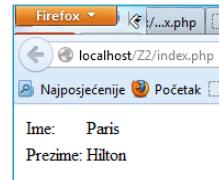
```
if ($korisnikID == 101){  
    echo "Dobro došli! <br>";  
    $prijavljen=true;  
}  
else {  
    echo "Pristup nije dozvoljen";  
    $prijevljen=false;  
}  
▶ umjesto otvorene vitičaste zagrada može se koristiti dvotočka (:), a  
tada cijela naredba završava ključnom riječi endif  
    if ($korisnikID == 101):  
        echo "Dobro došli! <br>";  
        $prijavljen=true;  
    else:  
        echo "Pristup nije dozvoljen";  
        $prijevljen=false;  
    endif;
```

5

5

## Ugradnja HTML-a u if ... else naredbu

```
<?php if ($korisnikID==101): ?>  
    <table>  
        <tr>  
            <td> Ime: </td> <td> Paris<td/>  
        </tr>  
        <tr>  
            <td>Prezime: </td> <td>Hilton</td>  
        </tr>  
    </table>  
<?php else: ?>  
    Molimo prijavite se u sustav!  
<?php endif; ?>
```



6

6

## Naredba If ... else

```
if ($x%2==0){  
    echo "Broj $x je paran";  
}  
else {  
    if ($x%7==0) {  
        echo "Broj $x je neparan i djeljiv sa 7";  
    }  
    else {  
        echo "Broj $x je neparan";  
    }  
}
```

- else se odnosi na najближи nezatvoren if

7

7

## Naredba If ... elseif ... else

```
if ($x%2==0){  
    echo "Broj $x je paran";  
}  
elseif ($x%7==0) {  
    echo "Broj $x je neparan i djeljiv sa 7";  
}  
else {  
    echo "Broj $x je neparan";  
}
```

8

8

## Naredba if i uvjetni operator

- ▶ Uvjetni operator može zamijeniti jednostavnu if...else naredbu

```
if (uvjet) {true_naredba} else {false_naredba}

    uvjet? true_naredba : false_naredba
if ($x%2==0){
    $status= "Broj $x je paran";
}
else {
    $status= "Broj $x je neparan";
}
($x%2==0)? $status= "Broj $x je paran" : $status= "Broj
    $x je neparan";
echo ($x%2==0)? "Broj $x je paran": "Broj $x je neparan";
```

9

9

## Naredba switch

- ▶ preglednija od if naredbe
- ▶ upotrebljiva kad se jedan izraz (vrijednost varijable ili rezultat neke operacije) testira na više različitih vrijednosti

```
switch($ocjena) {
    case 5:
        echo "odličan";
        break;
    case 4:
        echo "vrlo dobar";
        break;
    case 3:
        echo "dobar";
        break;
    case 2:
        echo "dovoljan";
        break;
    case 1:
        echo "nedovoljan";
        break;
    default:
        echo "neispravna ocjena";
        break;
}
```

10

10

## Naredba switch

- alternativna sintaksa umjesto { koristi dvotočku (:) a umjesto } koristi **endswitch**

```
switch($ocjena) :  
    case 5:  
        echo "odličan";  
        break;  
    case 4:  
        echo "vrlo dobar";  
        break;  
    case 3:  
        echo "dobar";  
        break;  
    case 2:  
        echo "dovoljan";  
        break;  
    case 1:  
        echo "nedovoljan";  
        break;  
    default:  
        echo "neispravna ocjena";  
        break;  
  
endswitch;
```

11

11

## Naredba switch

```
$broj=54;  
switch ($broj % 2){  
    case 0:  
        echo "broj je paran";  
    case 1:  
        echo "broj je neparan";  
}
```

ispis:

12

12

## Naredba while

- ▶ izvršava naredbu ako je uvjet true; nakon toga ponovno vrednuje uvjet – ako je uvjet i daje true ponovno se izvršava naredba, pa se vrednuje uvjet i tako dokle god je uvjet true

```
while (uvjet) naredba;
```

- ▶ za više naredbi koje treba ponavljati veći broj puta ako je uvjet ispunjen, koristi se blok naredbi koji se omeđuje s { na početku te s } na kraju bloka

```
while (uvjet) {
    naredba_1;
    naredba_2;

    ...;
    naredba_n;
}
```

13

13

## Naredba while

- ▶ umjesto otvorene vitičaste zgrade može se koristiti dvotočka (:), a tada cijela naredba završava ključnom riječi endwhile

```
while(uvjet) :
    naredba_1;
    naredba_2;

    ...;
    naredba_n;
endwhile;
```

14

14

## Naredba while

```
$suma=0;  
$i=1;  
while ($i<=100) {  
    $suma+=$i;  
    $i++;  
}  
echo "zbroj prvih 100 brojeva je $suma";
```

15

15

## Naredba while

- ▶ dodatnu kontrolu unutar petlje omogućuje korištenje naredbe **break** – izlazak iz petlje

```
$i=1;  
while ($i<=10) {  
    echo "$i <br>";  
    if ($i>7)  
        break;  
    $i++;  
}
```



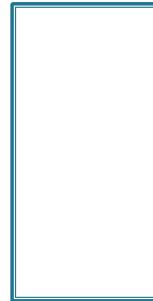
16

16

## Naredba while

- ▶ dodatnu kontrolu unutar petlje omogućuje korištenje naredbe **continue** – sve naredbe iza continue (a ispred kraja petlje) se preskaču, i ponovno se vrednuje uvjet

```
$i=0;  
while ($i<=10) {  
    $i++;  
    if ($i%2==0)  
        continue;  
    echo "$i <br>";  
  
}
```



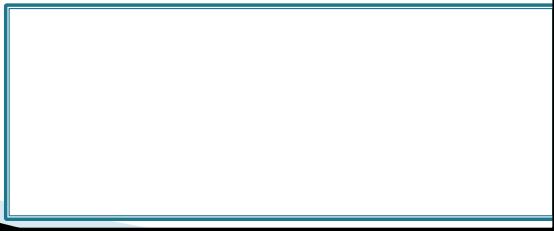
17

17

## Naredba while

- ▶ ugnježđivanje petlji

```
$i=1;  
$j=1;  
while ($i<=5) {  
    while ($j<=5) {  
        echo "$i x $j = " . $i*$j . "    ";  
        $j++;  
    }  
    echo "<br>";  
    $j=1;  
    $i++;  
}
```



18

## Naredba while

- ▶ break služi za izlazak samo iz petlje u kojoj se nalazi!

```
$i=1;  
$j=1;  
while ($i<=5){  
    while ($j<=5){  
        echo "$i x $j = " . $i*$j . " &nbsp &nbsp";  
        $j++;  
        if ($j>3)  
            break;  
    }  
    echo "<br>";  
    $j=1;  
    $i++;  
}
```

19

19

## Naredba while

- ▶ za izlazak iz većeg broja nivoa ugniježđenih petlji iza break može se dodati broj (npr. break 2;)

```
$i=1;  
$j=1;  
while ($i<=5){  
    while ($j<=5){  
        echo "$i x $j = " . $i*$j . " &nbsp &nbsp";  
        $j++;  
        if ($j>3)  
            break 2;  
    }  
    echo "<br>";  
    $j=1;  
    $i++;  
}
```

20

20

## Naredba while

- continue se ponaša kao i break (npr, u unutarnjoj petlji continue 2; skače na sljedeću iteraciju vanjske petlje)

```
while ($i<=5){  
    while ($j<=5){  
        if ($j==3){  
            echo "<br>";  
            $j=1;  
            $i++;  
            continue 2;  
        }  
        echo "$i x $j = " . $i*$j . "    ";  
        $j++;  
    }  
    echo "<br>";  
    $j=1;  
    $i++;  
}
```

21

21



## Naredba do while

- sve rečeno za while petlju vrijedi i za do while
- jedina razlika je što se uvjet ispituje na kraju petlje, što znači da će se do while petlja izvršiti barem jedanput
- nema alternativni sintaksni oblik (: i endwhile umjesto { i })

```
$i=0;  
do {  
    echo "$i <br>";  
    $i++;  
} while ($i<10);
```



22

22

## Naredba for

```
for (pocetni_izraz; uvjet; korak) {  
    naredbe;  
}  
▶ pocetni_izraz se računa samo jedanput i služi za postavljanje  
    brojača petlje na početnu vrijednost  
▶ dokle god je uvjet true, petlja će se izvršavati. uvjet se ispituje na  
    početku svakog ponavljanja  
▶ korak uvećanja (ili umanjenja) brojača računa se nakon što se  
    izvrši blok naredbi unutar petlje  
for ($i=0; $i<10;$i++) {  
    echo "$i <br>";  
}
```



23

23

## Naredba for

- ▶ napisati for petlju koja ispisuje brojeve od 20 prema 10

20  
19  
18  
17  
16  
15  
14  
13  
12  
11  
10

- ▶ bilo koji izraz u zaglavlju petlje može se izostaviti, a čak ih je  
 moguće izostaviti sve → beskonačna petlja

```
for(;;){  
    echo "vječni ispis";  
}
```

24

24

## Naredba for

- ▶ for petlja se može pretvoriti u while petlju:

```
for ($brojac=pocetna_vrij; $brojac<=kraj;$brojac++) {  
    naredbe;  
}
```

```
$brojac=pocetna_vrij;  
while ($brojac<=kraj) {  
    naredbe;  
    $brojac++;  
}
```

- ▶ pretvoriti while petlju u ekvivalentnu for petlju:

```
$suma=0;  
$i=1;  
while ($i<=100) {  
    $suma+=$i;  
    $i++;  
}  
echo "zbroj prvih 100 brojeva je $suma";
```

25

25



## P3: Funkcije

Kolegij: Programiranje u PHP  
Predavač: Dalibor Bužić

## Funkcije

- ▶ funkcija je skup naredbi koji čini jednu logičku cjelinu te izvršava određeni zadatak
  - naredbe u funkciji se pišu samo jednom, a mogu se pozvati neograničen broj puta
  - olakšano je ispravljanje grešaka, ali i održavanje prilikom naknadnih izmjena
  - moguće ih je relativno lagano uklopiti u druge projekte
  - mogu primati i po potrebi vraćati vrijednost(i)
- ▶ U PHP-u postoje ugrađene i korisničke funkcije

## Definiranje funkcije

```
function ime_funkcije([parametar[, ...]])  
{  
    naredbe  
}  
  
▶ naredbe mogu sadržavati HTML  
▶ imena funkcija nisu osjetljiva na veličinu slova,  
pa su pozivi IspisiDobavljace(),  
ISPISIDobavljace() i ispisidobavljace()  
jednako valjni  
▶ funkcije uobičajeno vraćaju neku vrijednost  
▶ vraćanje vrijednosti se ostvaruje naredbom  
    return izraz;
```

3

3

## Definiranje funkcije

```
function PretvoriTemperaturu($CelsiusTemp)  
{  
    $FahrenheitTemp=$CelsiusTemp*(9/5) + 32;  
    return $FahrenheitTemp;  
}  
  
ili  
  
function PretvoriTemperaturu($CelsiusTemp)  
{  
    return $CelsiusTemp*(9/5) + 32;  
}  
  
poziv funkcije:  
echo PretvoriTemperaturu(20);
```

4

4

## Opseg varijabli

- ▶ varijabla definirana unutar funkcije (što uključuje i parametre) lokalna je; tj. vidljiva je samo unutar te funkcije
- ▶ varijabla definirana izvan funkcije nije automatski vidljiva unutar funkcije!

5

5

## Opseg varijabli

```
$x=10;  
function uvecaj()  
{  
    $x+=5;  
}  
uvecaj();  
echo $x; //ispisuje
```



- ▶ da bi varijabla mogla biti korištena unutar funkcije, potrebno je unutar funkcije napisati naredbu oblika **global \$varijabla**; prije korištenja te varijable u funkciji

```
$x=10;  
function uvecaj()  
{  
    global $x;  
    $x+=5;  
}  
uvecaj();  
echo $x; //ispisuje
```



6

6

## Opseg varijabli

- ▶ u svakoj funkciji koja želi koristiti varijablu definiranu izvan funkcije, potrebno je navesti naredbu **global \$varijabla;**

```
$x=10;
function umanji()
{
    global $x;
    $x-=7;
}
function uvecaj ()
{
    global $x;
    $x+=5;
}
uvecaj();
echo $x; // ispisuje  
umanji();
echo $x; // ispisuje  
```

7

7

## Opseg varijabli

- ▶ izlaskom iz procedure lokalne varijable se brišu iz memorije

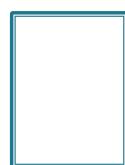
```
function ispisi()
{
    $x = 5;
    $x++;
    echo $x . "<br>";
}
for ($i=0; $i<5;$i++){
    ispisi();
}
```



- ▶ ukoliko bi se željelo sačuvati varijablu u memoriji nakon napuštanja procedure, potrebno je varijablu deklarirati pomoću ključne riječi **static**

- ▶ inicijalizacija statičke varijable obavlja se samo pri prvom ulasku u proceduru!

```
function ispisi()
{
    static $x = 5;
    $x++;
    echo $x . "<br>";
}
for ($i=0; $i<5;$i++){
    ispisi();
}
```



8

8

## Proslijedivanje vrijednošću ili referencom

- ▶ proslijedivanje varijable u proceduru podrazumijeva kopiranje njezine vrijednosti u parametar funkcije.
- ▶ to se naziva proslijedivanje vrijednošću
  - ▶ promjenom parametra ne mijenja se vrijednost varijable u pozivnom kodu

```
function uvecaj ($x)
{
    $x++;
}
$a=10;
uvecaj ($a);
echo $a; //ispisuje
```

- ▶ alternativni način slanja varijable u proceduru je davanje funkciji pristupa originalnoj varijabli
- ▶ to se naziva proslijedivanje referencom
- ▶ u parametarskoj listi ispred imena parametra stavlja se znak &
- ▶ promjena parametra ujedno znači promjenu originalne varijable u pozivnom kodu!

```
function uvecaj (&$x)
{
    $x++;
}
$a=10;
uvecaj ($a);
echo $a; //ispisuje
```

9

## Podrazumijevana vrijednost parametra

- ▶ parametar može imati podrazumijevanu vrijednost
- ▶ ako se prilikom poziva procedure izostavi varijabla, parametar procedure će poprimiti podrazumijevanu vrijednost
- ▶ parametri s podrazumijevanom vrijednošću se u zaglavlju procedure (u parametarskoj listi) stavljaju nakon što se navedu svi parametri bez podrazumijevane vrijednosti

```
function potenciraj($baza, $eksp = 2)
{
    $rez=1;
    for($i=1;$i<=$eksp;$i++){
        $rez=$rez*$baza;
    }
    return $rez;
}
echo potenciraj(4,3) . "<br>"; //ispisuje
echo potenciraj (4) . "<br>"; //ispisuje
```

- ▶ podrazumijevana vrijednost mora biti konstanta!

10

10

## Varijabilni broj parametara u funkciji

- ▶ ako programer prilikom izrade koda ne zna broj broj parametara koje će funkcija primiti, može definirati funkciju koja će moći primiti različit broj parametara
- ▶ to postiže tako da parametarsku listu u zaglavlju funkcije ostavi potpuno praznu
- ▶ PHP ima tri funkcije koje pomažu programeru u takvoj situaciji:
  - func\_get\_args() vraća sve parametre proslijedene funkciji u obliku niza
  - func\_num\_args() vraća broj parametara proslijedenih funkciji
  - func\_get\_arg(index) vraća parametar na određenoj poziciji niza

11

11

## Varijabilni broj parametara u funkciji

```
function zbroji()
{
    $zbroj=0;
    for($i=0;$i<func_num_args();$i++) {
        $zbroj+=func_get_arg($i);
    }
    return $zbroj;
}
$a=4; $b=10;
echo zbroji($a,$b) . "<br>";           //ispisuje  
echo zbroji(5,$a,20,$b,3) . "<br>"; //ispisuje  
```

12

12

## strict

```
<?php
function pomnozi(int $x, int $y) {
    return $x * $y;
}
echo pomnozi(6, "3 sata");
?>
```

- ▶ Od PHP-a verzije 7:

```
<?php declare(strict_types=1);
function pomnozi(int $x, int $y) {
    return $x * $y;
}
echo pomnozi(6, "3 sata");
?>
```

```
Fatal error: Uncaught TypeError: Argument 2 passed to pomnozi() must be of the type integer, string given, called in /wwwsvQxXR on line 5 and defined in /wwwsvQxXR:2 Stack trace: #0
```

13

13

## strict

- ▶ *declare(strict\_types=1);* mora se nalaziti u prvom redu PHP skripte
- ▶ Definirati se može i tip povratne vrijednosti

```
<?php declare(strict_types=1);
function pomnozi(int $x, int $y) : int {
    return $x * $y;
}

echo pomnozi(6,3);
?>
```

14

14

## Zadatak

- ▶ Napišite funkciju koja će ispisati svaki drugi broj u intervalu između dva cijela broja koja funkcija dobije u obliku parametara. Ako funkcija ne dobije drugi parametar, onda treba koristiti podrazumijevanju vrijednost 1000.

15

15

## Zadatak

- ▶ Napišite funkciju koja će prilikom prvog poziva vratiti 50, prilikom drugog poziva 49, prilikom trećeg 48, itd, a za sve pozive nakon pedesetog vratiti 0.

16

16

## Zadatak

- ▶ Napišite funkciju koja prima nepoznati broj ocjena vina. Funkcija treba izračunati i ispisati prosječnu ocjenu vina.



## P4: Nizovi

Kolegij: Programiranje u PHP  
Predavač: Dalibor Bužić

### Nizovi (matrice)

- ▶ integeri, brojevi s pomičnim zarezom, stringovi i Boolean su skalarni tipovi podataka (sadrže jednu vrijednost)
- ▶ niz (matrica, engl. *array*) je složeni tip podatka
- ▶ niz je uređena zbirka podataka, organizirana u obliku parova ključ – vrijednost
- ▶ nizovi su uobičajeni i korisni, a PHP posjeduje brojne ugrađene funkcije za rad s njima

## Vrste nizova

- ▶ nizovi u PHP-u mogu biti indeksirani i asocijativni
- ▶ ključevi indeksiranih nizova su cijeli brojevi koji počinju od 0
- ▶ asocijativni nizovi umjesto brojeva imaju stringove za ključeve; ponašaju se poput tablice s dva stupca (prvi stupac je ključ preko kojeg se pristupa vrijednosti u drugom stupcu)
- ▶ ključevi u obje vrste nizova moraju biti jedinstveni
- ▶ PHP interno pohranjuje obje vrste nizova kao asocijativne

3

3

## Indeksirani nizovi

- ▶ elementi niza se međusobno razlikuju po ključu (indeksu) koji je cjelobrojna vrijednost
- ▶ prvi element ima indeks 0, drugi element ima indeks 1, treći 2 itd.
- ▶ pojedinom elementu se pristupa tako da se iza naziva niza u uglatim zagradama navede indeks

<b>\$kolegij[0]</b>	"Programiranje baza podataka"
<b>\$kolegij[1]</b>	"Upravljanje računalnom sigurnosti"
<b>\$kolegij[2]</b>	"Sociologija informacijskog društva"
<b>\$kolegij[3]</b>	"Programiranje u PHP"
<b>\$kolegij[4]</b>	""Pouzdanost informacijskih sustava""

```
echo $kolegij[1];
// ispisuje
```

4

## Asocijativni nizovi

- ▶ elementi niza se međusobno razlikuju po ključu (indeksu) koji je string vrijednost
- ▶ pojedinom elementu se pristupa tako da se iza naziva niza u uglatim zagradama navede indeksna vrijednost
- ▶ sintaksa dopušta da se indeks ne stavlja unutar navodnika, no dobra je praksa indeksnu vrijednost staviti unutar navodnika

\$kolodvor['naziv']	"Autobusni kolodvor Zagreb"
\$kolodvor['adresa']	"Marina Držića 4"
\$kolodvor['telefon']	"060 313 333"

echo \$kolodvor[adresa];  
// ispisuje

5

5

## Pohranjivanje podataka u niz

- ▶ pohranjivanje vrijednosti u niz stvorit će niz ako on već ne postoji
- ▶ inicijalizacija indeksnog niza 1. način:

```
$kolegij[0]="Programiranje baza podataka";  
$kolegij[1]="Upravljanje računalnom sigurnosti";  
$kolegij[2]="Sociologija informacijskog društva";  
$kolegij[3]="Programiranje u PHP";  
$kolegij[4]="Pouzdanost informacijskih sustava";
```

- ▶ inicijalizacija indeksnog niza 2. način:

```
$kolegij=array("Programiranje baza podataka",  
"Upravljanje računalnom sigurnosti",  
"Sociologija informacijskog društva",  
"Programiranje u PHP", "Pouzdanost informacijskih sustava");
```

6

6

## Pohranjivanje podataka u niz

- ▶ inicijalizacija indeksnog niza 3. način:

```
$kolegij=[ "Programiranje baza podataka",  
          "Upravljanje računalnom sigurnosti",  
          "Sociologija informacijskog društva",  
          "Programiranje u PHP", "Pouzdanost informacijskih sustava"];
```

7

7

## Pohranjivanje podataka u niz

- ▶ inicijalizacija asocijativnog niza 1. način:

```
$kolodvor['naziv']="Autobusni kolodvor Zagreb";  
$kolodvor['adresa']="Marina Držića 4";  
$kolodvor['telefon']="060 313 333";
```

- ▶ inicijalizacija asocijativnog niza 2. način:

```
$kolodvor=array(  
    'naziv'=>"Autobusni kolodvor Zagreb",  
    'adresa'=>"Marina Držića 4",  
    'telefon'=>"060 313 333"  
) ;
```

- ▶ inicijalizacija asocijativnog niza 3. način:

```
$kolodvor=[ 'naziv'=>"Autobusni kolodvor Zagreb",  
            'adresa'=>"Marina Držića 4", 'telefon'=>"060 313 333"];
```

8

8

## Dodavanje novog elementa u niz

- ▶ dodavanje novog elementa na kraj postojećeg indeksnog niza:

```
$pisci=array("Krleža", "Ujević", "Marinković");
```

```
$pisci[]="Nazor";
```

- ▶ dodavanje novog elementa na kraj postojećeg asocijativnog niza:

```
$osoba=array('ime'=>"Častislav",
'zanimanje'=>"čuvar državnog pečata");
```

```
$osoba['država']="Hrvatska";
```

9

9

## Stvaranje niza sa slijednim vrijednostima

- ▶ funkcija range() stvara niz slijednih brojčanih ili znakovnih vrijednosti

```
$brojevi=range(4,8);
$slova=range('i','m');
$odveceg=range(8,4);
$viseslova=range('iva','mia')
```

10

10

## Višedimenzionalni nizovi

- ▶ niz može umjesto skalarnih elemenata sadržavati druge nizove
- ▶ u takvom slučaju govorimo o višedimenzionalnim nizovima

```
$niz1=array(1,3,5,7);  
$niz2=array(0,2,4,8);  
$niz3=array(9,6,20,10);  
$multiniz=array($niz1,$niz2,$niz3);
```

```
echo $multiniz[2][3]; //ispisuje 
```

- ▶ nizovi-elementi se mogu međusobno razlikovati po broju elemenata

11

11

## Funkcije za rad s nizovima

- ▶ count() i sizeof() vraćaju broj elemenata u nizu

```
$niz3=array(9,6,20,10);  
echo count($niz3); //ispisuje 
```
- ▶ list() kopira vrijednosti elemenata u varijable
  - često se koristi prilikom dohvatanja vrijednosti iz baze podataka
  - ako je broj elemenata u nizu veći od broja varijabli, višak elemenata se ignorira
  - ako je broj elemenata u nizu manji od broja varijabli, višak varijabli poprima vrijednost NULL
  - s dva uzastopna zareza u listi varijabli, preskače se element niza

12

12

## Funkcije za rad s nizovima

```
$U2=array ("Bono", "Adam", "Larry","The Edge");
list($vokal, $bas, $bubnjevi,$gitara)=$U2;
//  
list($v, $b)=$U2;
//  
list($vokal, $bas, $bubnjevi,$gitara, $harmonika)=$U2;
if(is_null($harmonika))
    echo "harmonika je null <br>";
list($prvi, , , $drugi)=$U2;
echo $prvi . " " . $drugi;
//ispis:
//  
//
```

13

13

## Funkcije za rad s nizovima

- ▶ array\_slice() stvara novi niz na temelju postojećeg niza

```
$podniz=array_slice($niz, pocetak, broj_el)
```

```
$sastav=array("Bono", "Adam", "Larry","The Edge");
$novi=array_slice($sastav,1,2);
```

- ▶ array\_keys() vraća niz koji se sastoji samo od ključeva u nizu

```
$crtic=array('macak'=>"Tom", 'mis'=>"Jerry",'razlog'=>"sir");
$tko=array_keys($crtic); //elementi su:  
$niz1=range(10,15);
$tko=array_keys($niz1); //elementi su:
```

14

14

## Funkcije za rad s nizovima

- ▶ array\_splice() uklanja ili dodaje te optionalno stvara novi niz sastavljen od uklonjenih elemenata postojećeg niza

```
$novi=array_splice($niz, pocetak, [broj_el[,  
novi_el]]);  
  
$zaprta=array("akcija","krimic","drama","western","triler",  
"horor");  
$novi=array_splice($zaprta, 1,3);  
//$novi je  
// ali $zaprta
```

- ▶ nije nužno stvarati novi niz od uklonjenih elemenata:

```
$zaprta=array("akcija","krimic","drama","western","triler",  
"horor");  
array_splice($zaprta, 2); //
```

15

15

## Funkcije za rad s nizovima

- ▶ dodavanje novih elemenata na mjesto uklonjenih:

```
$zaprta=array("akcija","krimic","drama","western","triler",  
"horor");  
$dodatni=array("mjuzikl", "komedija");  
array_splice($zaprta,2,3,$dodatni);  
//$zaprta je sad:  
//
```

- ▶ broj uklonjenih i novoubačenih elemenata ne mora se poklapati

16

16

## Funkcije za rad s nizovima

- ▶ extract() stvara varijable iz niza

```
$crtic=array('macak'=>"Tom", 'mis'=>"Jerry", 'razlog'=>"sir");
extract($crtic);
echo $macak . " ". $mis . " " . $razlog;
//ispisuje
```

- ▶ opcija EXTR\_PREFIX\_ALL omogućuje stvaranje jedinstvenih varijabli dodavanjem prefiksa imenu varijabi

```
$crtic=array('macak'=>"Tom", 'mis'=>"Jerry", 'razlog'=>"sir");
extract($crtic, EXTR_PREFIX_ALL,"mgm");
echo $mgm_macak . " ". $mgm_mis . " " . $mgm_razlog;
```

17

17

## Funkcije za rad s nizovima

- ▶ compact() stvara niz iz varijabli

```
$naziv="čokolada";
$kolicina=100;
$cijena=14.99;
$artikl=compact("naziv", "kolicina","cijena");
echo $artikl['naziv'] . " ". $artikl['cijena'] . " " .
$artikl['kolicina'];

//ispisuje
```

18

18

## Petlja foreach

- ▶ petlja foreach služi za 'prolazak' kroz sve elemente niza
- ▶ naredbe u petlji se izvršavaju onoliko puta koliko niz ima elemenata

```
$zanol=array("akcija","krimic","drama","triler","horor");  
foreach($zanol as $vrijednost){  
    echo "$vrijednost <br>";  
}
```

- ▶ ako se želi pristupiti i ključu (kod asocijativnih nizova) onda se to može napraviti na slijedeći način:

```
$crtic=array('macak'=>"Tom", 'mis'=>"Jerry", 'razlog'=>"sir");  
foreach($crtic as $kljuc =>$vrijednost){  
    echo "$kljuc : $vrijednost <br>";  
}
```

19

19

## Korištenje for petlje s nizovima

- ▶ for petlja može biti zgodna za prolazak kroz indeksirane nizove čiji ključevi (indeksi) počinju od nule i slijedno se uvećavaju
- ▶ za razliku od foreach petlje, for radi s originalnim nizom (foreach s kopijom niza)

```
$zanol=array("akcija","krimic","drama","triler","horor");  
$koliko=count($zanol);  
for($i=0; $i<$koliko; $i++){  
    echo "$zanol[$i] <br>";  
}
```

20

20

## Pozivanje funkcije za svaki element niza

- array\_walk() omogućuje pozivanje korisničke funkcije po jednom za svaki element niza

```
function IspisiRedak($vrij,$kljuc,$boja)
{
    echo "<tr>\n<td bgcolor=\"{$boja}\">{$vrij}</td>";
    echo "<td bgcolor=\"{$boja}\">{$kljuc}</td>\n</tr>\n";
}
$crtic=array('macak'=>"Tom", 'mis'=>"Jerry", 'razlog'=>"sir");
echo "<table border=\"1\">";
array_walk($crtic, "IspisiRedak", "teal");
echo "</table>";
```

Tom	macak
Jerry	mis
sir	razlog

21

21

## Ispitivanje vrijednosti

- in\_array() funkcija vraća true ako traženi element postoji u nizu, a false vraća ako traženog elementa nema u nizu

```
$U2=array("Bono", "Adam", "Larry", "The Edge");
$imaLarrija=in_array("Larry", $U2); // 
$imaRinga=in_array("Ringo", $U2); // 
```

22

22

# Sortiranje

- ▶ u PHP-u su tri načina sortiranja:
  - sortiranje po ključevima
  - sortiranje po vrijednostima bez mijenjanja ključeva
  - sortiranje po vrijednostima uz mijenjanje ključeva
- ▶ pored toga, sortiranje može biti rastuće, padajuće ili pak korisnički definirano

Način	Rastuće	Padajuće	Korisnički redoslijed
Sortiranje po vrijednostima uz promjenu ključeva	sort()	rsort()	usort()
Sortiranje po vrijednostima	asort()	arsort()	uasort()
Sortiranje po ključevima	ksort()	krsort()	uksort()

23

23

# Sortiranje

- ▶ sort(), rsort() i usort() su namijenjene indeksiranim nizovima → mijenjaju vrijednosti ključeva u skladu s redoslijedom

```
$U2=array("The Edge","Bono", "Adam", "Larry");
echo "Prije sortiranja: <br>";
print_r($U2);
echo "<br>Nakon sortiranja: <br>";
sort($U2);
print_r($U2);
```

Prije sortiranja:  
Array ( [0] => The Edge [1] => Bono [2] => Adam [3] => Larry )  
Nakon sortiranja:  
Array ( [0] => Adam [1] => Bono [2] => Larry [3] => The Edge )

- ▶ korištenjem  
asort(\$U2);  
dobije se ovo→

Prije sortiranja:  
Array ( [0] => The Edge [1] => Bono [2] => Adam [3] => Larry )  
Nakon sortiranja:  
Array ( [2] => Adam [1] => Bono [3] => Larry [0] => The Edge )

24

24

## Sortiranje

```
$komentatori=array('vanja'=>45, 'mia'=>62, 'tanja'=>48,  
'miky'=>12);  
arsort($komentatori); //silazno sortiranje po vrijednostima  
echo "<table border=\"1\">";  
foreach($komentatori as $korisnik => $koliko){  
echo ("<tr><td>{$korisnik}</td><td>{$koliko}</td></tr>\n");  
}  
echo "</table>";
```

mia	62
tanja	48
vanja	45
miky	12

25

25

## Istovremeno sortiranje više nizova

```
$student=array("Pero", "Tugomir", "Andreja", "Izabela");  
$kolegij=array("Baze", "Mreže", "Mreže", "PHP");  
$ocjena=array(4,2,5,3);  
array_multisort($ocjena, SORT_DESC, $student, SORT_ASC,  
$kolegij);  
for ($i=0;$i<count($student);$i++){  
    echo "{$student[$i]},  
{$kolegij[$i]},{$ocjena[$i]}<br>";  
}
```

Andreja, Mreže, 5  
Pero, Baze, 4  
Izabela, PHP, 3  
Tugomir, Mreže, 2

- ▶ prvi element svakog niza predstavlja podatke o Peri, drugi element svakog niza podatke o Tugomiru itd.
- ▶ funkcija array\_multisort() preslaguje elemente nizova uz očuvanje cjelovitosti zapisa
- ▶ opcija SORT\_ASC znači uzlazno sortiranje, a SORT\_DESC silazno sortiranje

26

26

## ...još o sortiranju

- ▶ sortiranje ne vodi računa o 'normalnom' redoslijedu kod podataka koji sadrže i slova i brojeve
- ▶ npr. datoteke slika20.jpg, slika4.jpg i slika2.jpg dosad izloženim funkcijama za sortiranje napravile bi poredak slika2.jpg, slika20.jpg, slika4.jpg
- ▶ funkcije natsort() i natcasesort() obavljaju prirodno sortiranje (slika2.jpg, slika4.jpg, slika20.jpg)
- ▶ funkcija array\_reverse() stvara obrnuti niz
- ▶ funkcija shuffle() mijenja redoslijed elemenata u slučajni

```
$U2=array("The Edge", "Bono", "Adam", "Larry");  
shuffle($U2);  
print_r($U2);
```

```
Array ( [0] => Bono [1] => Adam [2] => Larry [3] => The Edge )
```

27

27

## Operacije nad cijelim nizom

- ▶ funkcija array\_sum() zbraja sve elemente niza
- ▶ funkcija array\_merge() spaja dva ili više nizova
- ▶ funkcija array\_diff() vraća vrijednosti koje se nalaze u prvom nizu, a nema ih niti u jednom od preostalih nizova

```
$racuni=array(134.00, 78.00, 12.00, 100.58);  
echo array_sum($racuni) . "<br>";  
$boje1=array("crna", "plava");  
$boje2=array("crvena", "plava", "zelena");  
$sveboje=array_merge($boje1,$boje2);  
print_r($sveboje);  
$b1=array("bijela", "crna", "plava", "smeda", "žuta");  
$b2=array("narančasta", "bijela", "ljubičasta", "žuta");  
$b3=array("zelena", "žuta", "smeda");  
$razlika=array_diff($b1,$b2,$b3);  
echo "<br>";  
print_r($razlika);
```

28

28

## Skupovne operacije

- ▶ skupovna operacija unije realizira se tako da se dva niza spoje funkcijom array\_merge(), a zatim uklone duplikati pomoću funkcije array\_unique()
- ▶ za operaciju presjeka postoji funkcija array\_intersect()
- ▶ funkcija array\_diff() vraća vrijednosti koje se nalaze u prvom nizu, a nema ih niti u jednom od preostalih nizova

```
$b1=array("bijela","crna","plava","smeda","žuta");  
$b3=array("zelena","žuta","smeda");  
$unija=array_merge($b1,$b3);  
$unija=array_unique($unija);  
print_r($unija);  
echo "<br>";  
$presjek=array_intersect($b1,$b3);  
print_r($presjek);
```



## P5: Rad s nizovima znakova

Kolegij: Programiranje u PHP  
Predavač: Dalibor Bužić

### Stringovi

- ▶ string = niz znakova (slova, brojevi, interpunkcija,...)
- ▶ živimo u multimedijalnom dobu, ali tekst je i dalje dominantan podatkovni format na web stranicama
- ▶ unos podataka preko forme, vraćanje rezultata upita nad bazom, komunikacija među korisnicima... stringovi su sveprisutni
- ▶ PHP je bogat po pitanju funkcija za rad s nizovima znakova

## String konstante

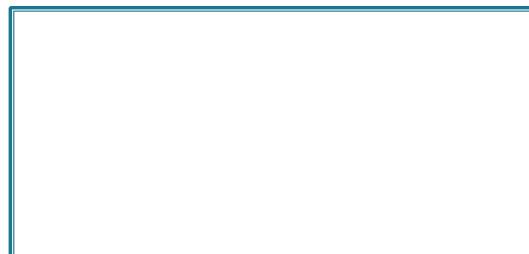
- ▶ stringovi se mogu pohraniti:
  - unutar jednostrukih navodnika
  - unutar dvostrukih navodnika
  - u here document (heredoc) formatu
  - u now document (nowdoc) formatu
- ▶ ova četiri načina se ponašaju različito!
- ▶ ako se string stavi unutar dvostrukih navodnika ili heredoc-a, moguće je unutar njega prikazati vrijednosti varijabli
- ▶ zamjena imena varijable njezinom vrijednošću naziva se interpolacija (umetanje)
- ▶ umetanje se može napraviti jednostavno navođenjem imena varijable ili obgrijivanjem imena varijable vitičastim zagradama unutar stringa

3

3

## Umetanje varijabli u string

```
$ona='Victoria';
$on='David';
echo '$ona i $on <br>';
echo "$ona i {$on} <br>";
$x=100;
echo "You are $xth winner! <br>";
echo "You are {$x}th winner! <br>";
$str="$ona";
echo $str;
```



4

4

## Escape znakovi

- ▶ string unutar jednostrukih navodnika prepoznaće samo \' i \\ kao escape znakove (omogućuju upotrebu jednostrukog navodnika i \ unutar stringa)
- ▶ unutar dvostrukih moguće je koristiti:
  - \" za dvostruki navodnik
  - \n za novi red
  - \r za povratak na početak reda
  - \t za tab
  - \\$ za znak dolara
  - ...

```
$p='Sinead O\' Connor pjeva \'Nothing compares 2U\'';
echo $p . "\n";
$s="Arnold je rekao: \n \"I'll be back!\"";
echo $s . "<br>";
```

5

## Heredoc

- ▶ heredoc omogućuje pohranjivanje višelinjskog teksta
- ▶ heredoc počinje s <<< ime
- ▶ string započinje u sljedećem redu, a završava kad dođe do linije koda koja sadrži SAMO ime i ;

```
$oblak=<<<citat
I svak je išo svojim putem:
Za vlašću, zlatom il za hljebom,
A on -- krvareći ljepotu --
Svojim nebom.
citat;
echo $oblak;
```

I svak je išo svojim putem:
Za vlašću, zlatom il za hljebom,
A on -- krvareći ljepotu --
Svojim nebom.

- ▶ u web pregledniku sav tekst može biti u jednom redu, no u source-u (CTRL+U u Firefoxu, ili Alt+V+C u IE 8+) je vidljiv efekt

6

6

## Heredoc

```
$page_title = 'Menu';
$meat = 'pork';
$vegetable = 'bean sprout';
print <<<MENU
<html>
<head><title>$page_title</title></head>
<body>
<ul>
<li> Barbecued $meat
<li> Sliced $meat
<li> Braised $meat with $vegetable
</ul>
</body>
</html>
MENU;
```

7

7

## Nowdoc

- ▶ uveden je u PHP od verzije 5.3.
- ▶ sličan heredoc-u, no NEMA mogućnost zamjene varijable vrijednostima
- ▶ definicija nowdoc-a se od heredoc-a razlikuje samo po tome što ime na početku definicije treba biti unutar jednostrukih navodnika
- ▶ zgodan je za pohranu PHP koda

```
$naredbe=<<<'kod'
$ona='Victoria';
$on='David';
echo '$ona i $on <br>';
echo "$ona i {$on} <br>";
kod;
echo $naredbe;
```

```
$ona='Victoria';
$on='David';
echo 'Sona i $on <br>';
echo "$ona i {$on} <br>";
```

8

8

## Naredbe za ispis stringova

- ▶ PHP ima četiri naredbe za ispis znakovnih nizova u web pregledniku:
  - echo može ispisati više vrijednosti
  - print – slična echo naredbi
  - printf() – moguće je formatirati ispis prema predlošku
  - print\_r() – ispisuje sadržaj npr. nizova ili objekata u čitljivom obliku, korisna kod ispravljanja grešaka

9

9

## Echo i print

- ▶ echo, iako sliči i u većini slučajeva se ponaša, nije funkcija već jezični konstrukt
- ▶ zato se ono što ispisuje može, ali i ne mora staviti unutar zagrade
- ▶ ako ispisuje više vrijednosti, zgrade se ne smiju koristiti
- ▶ echo je zasebna naredba, ne može se koristiti npr. u sklopu uvjeta petlje ili naredbe grananja
- ▶ print se ponaša vrlo slično naredbi echo

10

10

## printf()

- ▶ omogućuje formatiranje ispisnih vrijednosti
- ▶ unutar teksta moguće je umetati formatirane parametre koji počinju znakom %, optionalno modifikatorom, a završavaju specifikatorom tipa
- ▶ modifikatori mogu biti:
  - popunjavanje  $sn$  od kojih je  $s$  znak kojim se popunjuje, a  $n$  ukupan broj znakova ispisane vrijednosti.
  - znak – i + → na znakovne podatke minus znači lijevo poravnanje (podrazumijevano je desno), a na brojčane + znači da će se predznak ispisivati i kod pozitivnih brojeva
  - minimalna dužina
  - kod decimalnih brojeva .n znači da se iza decimalne točke prikazuje n decimala

11

11

## printf()

- ▶ neki specifikatori tipa:

znak	značenje
%	prikazuje znak %
b	argument je integer, prikazuje se u binarnom obliku
c	argument je integer, prikazuje se u znakovnom obliku
d	argument je integer, prikazuje se u decimalnom obliku
e, E	argument je decimalan, prikazuje se u ekspon. obliku
F	argument je decimalan, prikazuje se u decimalnom obliku
o	argument je integer, prikazuje se u oktalnom obliku
s	argument je string , prikazuje se u string obliku
u	argument je integer, prikazuje se u decimalnom obliku
x, X	argument je integer, prikazuje se u heksadecimalnom obliku

12

12

## printf() primjeri

```
$a=245;  
printf("U binarnom: %b, u oktalnom: %o\n", $a, $a);  
printf("U heksadecimalnom: %x \n", $a);  
printf("%d\n", $a);  
printf("%.2d\n", $a);  
printf("%.2g\n", $a);  
$t1 = 'lovac';  
printf("|%s|\n", $t1);  
printf("|%12s|\n", $t1);  
printf("|%-12s|\n", $t1);  
printf("|%012s|\n", $t1);  
printf("|%'#12s|\n", $t1);  
  
U binarnom: 11110101, u oktalnom: 365  
U heksadecimalnom: f5  
245  
245  
2.4e+2  
|lovac|  
|      lovac|  
|lovac     |  
|0000000lovac|  
|#####lovac|
```

13

13

## sprintf()

- sprintf() je funkcija sa mogućnostima kao i printf(), ali ona ne ispisuje string, već ga samo stvara, pa string možemo pohraniti u varijablu (ili datoteku) radi kasnijeg korištenja

```
$koliko = 10;  
$mjesto = 'košara';  
$tekst = '%s sadrži %d jabuka';  
echo sprintf($tekst, $mjesto, $koliko);  
//ispisuje
```

14

14

## print\_r() i var\_dump()

```
$crtic=array('macak'=>"Tom", 'mis'=>"Jerry",'razlog'=>"sir");
print_r($crtic);
// ispis:
    Array
    (
        [macak] => Tom
        [mis] => Jerry
        [razlog] => sir
    )

print_r(true); //ispisuje "1"
print_r(false); //ispisuje ""
print_r(null); //ispisuje ""
```

15

15

## print\_r() i var\_dump()

- ▶ zbog toga što print\_r() ne daje suvisao ispis za Boolean vrijednosti i NULL, kod ispravljanja pogrešaka prikladnija je var\_dump() funkcija jer ispisuje PHP vrijednosti u čovjeku čitljivom obliku

```
$artikl = array('naziv'=>"čokolada", 'cijena'=>12.98 );
var_dump(true);
var_dump(false);
var_dump(null);
var_dump($artikl);
```

```
boolean true
boolean false
null
array (size=2)
  'naziv' => string 'čokolada' (length=9)
  'cijena' => float 12.98
```

16

16

## Pristup pojedinačnim znakovima

- ▶ pojedinačnom znaku u stringu može se pristupiti sa `$strVar[$i]` gdje je `$strVar` naziv string varijable, a `$i` indeks znaka (prvi znak ima indeks 0)
- ▶ funkcija `strlen()` vraća broj znakova u nizu

```
$rijeka='Kupa';
$koliko=strlen($rijeka);
for($i=0;$i<$koliko;$i++){
    echo "$i. znak je: ". $rijeka[$i] .      0. znak je: K
    "<br/>";                                1. znak je: u
}                                              2. znak je: p
                                                3. znak je: a
```

17

17

## Pročišćavanje stringova

- ▶ funkcija `trim(niz[, znakovi])` uklanja vodeća i završna prazna mjesta u nizu što osim razmaka uključuje i "\t" (tab), "\n" (newline), "\r" (carriage return), "\0" (nul-bajt), "\xOB" (vertikalni tab)
- ▶ po želji može se ukloniti neki drugi skup znakova ako se navede opcionalni argument funkcije trim()
- ▶ `ltrim()` uklanja neželjene znakove s početka stringa
- ▶ `rtrim()` uklanja neželjene znakove s kraja stringa

```
$tekst=" lovac i vuk \n \t \n";
echo trim($tekst);
$tekst="##lovac i vuk #";
echo trim($tekst,"#");
// ispisuje:
```

18

18

## Pročišćavanje stringova

- ▶ *strtolower()* sva velika slova pretvara u mala
- ▶ *strtoupper()* sva velika slova pretvara u mala
- ▶ *ucfirst()* samo prvo slovo u stringu pretvara u veliko
- ▶ *ucwords()* svakoj riječi u stringu pretvara prvo slovo u veliko

```
$bajka="iviCA i maRICA";
echo strtolower($bajka) . "<br>";
echo strtoupper($bajka) . "<br>";
echo ucfirst($bajka) . "<br>";
echo ucwords($bajka) . "<br>";
```

```
ivica i marica
IVICA I MARICA
IviCA i maRICA
IviCA I MaRICA
```

19

19

## htmlentities()

- ▶ *htmlentities()* zamjenjuje specijalne znakove s HTML prikazom:
  - & postaje &amp
  - " postaje &quot
  - ' postaje #039
  - < postaje &lt, a > postaje &gt

```
$tekst=<<<upit
<p>"Što ćeš s njime?" pijet'o pita.</p>
upit;
//echo $tekst . "<br>";
echo htmlentities($tekst) . "<br>";

&lt;p&gt;&quot;&Scaron;to će&scaron; s njime?&quot; pijet'o
pita.&lt;/p&gt;
```

20

20

## strip\_tags()

- ▶ *strip\_tags()* uklanja HTML tagove
- ▶ kao drugi argument može se navesti tag (samo otvarajući tag) koji želimo ostaviti

```
$odgovor='<p><b>Prodat ču ga</b> za <i>tri zrna žita</i></p>';
echo $odgovor . "<br>";
echo strip_tags($odgovor) . "<br>";
echo strip_tags($odgovor,'<b>') . "<br>";
```

**Prodat ču ga za tri zrna žita**

Prodat ču ga za tri zrna žita  
**Prodat ču ga za tri zrna žita**

21

21

## get\_meta\_tags() i urlencode()

- ▶ *get\_meta\_tags()* vraća niz meta tagova iz HTML stranice

```
$meta=get_meta_tags('http://www.carnet.hr/');
print_r($meta);
Array
(
    [keywords] => CARNet, carnet, hrvatska, akademska, istrazivacka, mreza
    [language] => Hrvatski
)
```

- ▶ *urlencode()* obavlja pretvorbu u URL enkodiranje
- ▶ nedozvoljeni znak razmaka pretvara u znak +

```
$osnovniUrl='https://www.google.com/#q=';
$upit='planine u hrvatskoj';
$url=$osnovniUrl . urlencode($upit);
echo $url;
// ispisuje https://www.google.com/#q=planine+u+hrvatskoj
```

22

22

## Usporedba stringova

- ▶ operator == pretvara ne-string operanda u string vrijednost, pa su 54 i '54' jednaki
- ▶ operator === ne obavlja pretvorbu, pa ako uspoređuje ne-string operand sa string operandom, kao rezultat daje false
- ▶ operatori usporedbe primjenjivi su na stringove, vrijedi da je 'Antonija' > 'Anamarija'
- ▶ međutim, ako je jedan operand broj, onda se string operand pretvara u broj i iznosi 0 pa je 23 > "kozlić"
- ▶ `strcmp(string1, string2)` uspoređuje na (ne)jednakost
  - vraća vrijednost manju od 0 ako string1 dolazi prije od string2
  - vraća vrijednost veću od 0 ako string1 dolazi nakon string2
  - vraća vrijednost 0 ako su string1 i string2 jednaki

23

23

## Usporedba stringova

- ▶ `strcasecmp()` je varijanta od `strcmp()`, a ona pretvara argumente u mala slova prije nego ih usporedi
- ▶ `strnatcmp()` i `strnatcasecmp()` su zgodne za usporedbu stringova koji sadrže numerički dio: string dio poredaju tekstualni dio po abecedi, a numerički prema brojevnom pravcu
- ▶ sličnost stringova može se ispitati pomoću sljedećih funkcija:
  - `soundex()`
  - `metaphone()`
  - `similar_text()`
  - `levenshtein()`

24

24

## Usporedba stringova

```
$poznat='bread';
$upitan='breath';
if(soundex($poznat)==soundex($upitan)){
    echo "za soundex $poznat zvuči kao $upitan <br>";
}
else {
    echo "za soundex $poznat i $upitan ne zvuče slično
<br>";
}
if(metaphone($poznat)==metaphone($upitan)){
    echo "za metaphone $poznat zvuči kao $upitan <br>";
}
else {
    echo "za metaphone $poznat i $upitan ne zvuče slično
<br>";
}
```

za soundex bread zvuči kao breath  
za metaphone bread i breath ne zvuče slično

25

25

## Usporedba stringova

- ▶ similar\_text() broji zajedničke znakove

```
$prvi="Malinska";
$drugi="Makarska";
$zajedn=similar_text($prvi, $drugi,$postotak);
echo "Zajedničkih je $zajedn znakova, što iznosi $postotak%";
//ispisuje: Zajedničkih je 5 znakova, što iznosi 62.5%
```

- ▶ Levenshtein algoritam računa sličnost dvaju stringova po tome koliko znakova treba oduzeti, dodati ili zamijeniti kako bi dva stringa bila jednaka

```
$prvi="vuk";
$drugi="koka";
$slicno=levenshtein($prvi, $drugi);
echo "$slicno";      // $slicno=3
```

26

26

## Operacije sa stringovima

- ▶ substr(string, pocetak [,broj\_zn]) izvlači podniz, pocetak je pozicija od koje počinje i ide do kraja ako nije naveden broj\_zn
- ▶ substr\_count(veciNiz, manjiNiz) broji koliko se puta manji niz nalazi u većem

```
$jmbg="1905983315201";
$god=substr($jmbg,4,3);
echo "Godina rođenja: $god <br>";
$stih="Breze su vre čisto žute";
echo substr($stih,12). "<br>";
$stih="cvrči cvrči cvrčak na čvoru crne smrče";
echo substr_count($stih,'rč'). "<br>";
```

27

27

## Operacije sa stringovima

- ▶ substr\_replace(orig, novi, pocetak [,broj\_zn]) zamjenjuje dio stringa orig sa stringom novi, pocetak je pozicija od koje počinje i ide do kraja ako nije naveden broj\_zn

```
$sut="sutra je smak svijeta";
echo substr_replace($sut,"neće biti",6,2) . "<br>";
echo substr_replace($sut,"novi ",9,0) . "<br>";
echo substr_replace($sut,"",0,8) . "<br>";
```

28

28

# Operacije sa stringovima

- ▶ strrev(string) okreće niz
  - ▶ str\_repeat(string, n) string ponavlja n puta
  - ▶ str\_pad(string, length[, with [, pad\_type]])
    - length označava broj mesta
    - with s kojim znakom
    - pad\_type – popunjavanje podrazumijevano zdesna (STR\_PAD\_RIGHT), popunjavanje slijeva (STR\_PAD\_LEFT), ili centrirano (STR\_PAD\_BOTH),
- ```
$voda="Evian";
echo strrev($voda) . "<br>";
echo str_repeat("^_/_", 5) . "<br>";
$pisac="Miroslav Krleža";
echo "|" . str_pad($pisac,25,'.',STR_PAD_RIGHT) . "|<br>";
echo "|" . str_pad($pisac,25,'.',STR_PAD_LEFT) . "|<br>";
echo "|" . str_pad($pisac,25,'.',STR_PAD_BOTH) . "|<br>";
```

```
naive
^_/_/^_/_/^_/_/^_/_/^_/_/^_
|Miroslav Krleža.....|
|.....Miroslav Krleža|
|....Miroslav Krleža....|
```

29

# Operacije sa stringovima

- ▶ \$array=explode(separator, string [, limit]);
  - separator je znak koji će odjeljivati komponente
  - limit je najveći broj elemenata novonastalog niza array (ako je limit veći od mogućeg broja elemenata, onda će zadnji element sadržavati sve preostale vrijednosti)
- ▶ \$string=implode(separator, array);
  - radi suprotno od funkcije explode()
  - alias za implode() je join()

```
$U2="Bono,Adam,Larry,The Edge";
$niz=explode(',',$U2);
print_r($niz);
Array
(
    [0] => Bono
    [1] => Adam
    [2] => Larry
    [3] => The Edge
)

$komentatori=array('sanja', 'mia', 'tanja', 'miky');
$str=implode(',',$komentatori);
echo $str; //ispisuje sanja,mia,tanja,miky
```

30

30

## Operacije sa stringovima

- ▶ `strtok(string,separator);`
  - kod prvog poziva obavezno se navodi string
  - kod slijedećih poziva samo separator
- ▶ `$array=sscanf(string, template); //ili:`
  - `$count=sscanf(string, template [, var1...]);`
  - radi dekompoziciju stringa prema predlošku printf() funkcije
  - bez opcija argumenata vraća niz

```
$U2="Bono Adam Larry The Edge";
$clan=strtok($U2, ' ');
while($clan !== false){
    echo "$clan <br>";
    $clan=strtok(' ');
}
```

Bono  
Adam  
Larry  
The  
Edge

```
$str="Tin Ujević (1891)";
$niz=sscanf($str,"%s %s (%d)");
print_r($niz);
```

Array  
(  
[0] => Tin  
[1] => Ujević  
[2] => 1891  
)

31

## Operacije sa stringovima

- ▶ `strpos(large_string, small_string);`
  - vraća poziciju prvog pojavljivanja malog stringa u velikom
  - `strrpos()` vraća poziciju zadnjeg pojavljivanja
- ▶ `strstr(large_string, small_string);`
  - nalazi prvo pojavljivanje malog stringa u većem i vraća sve od tog prvog pojavljivanja do kraja
- ▶ `parse_url(url);`
  - vraća niz čiji su elementi komponente url-a

```
$tin="Ne boj se! nisi sam! ima i drugih nego ti";
echo strpos($tin,'! ') . "<br>"; //ispisuje 9
echo strstr($tin,'! ') . "<br>";
// ispisuje: ! nisi sam! ima i drugih nego ti
```

```
$url="https://www.google.com/#q=planine+u+hrvatskoj";
$niz= parse_url($url);
print_r($niz);
```

Array  
(  
[scheme] => https  
[host] => www.google.com  
[path] => /  
[fragment] => q=planine+u+hrvatskoj  
)

32

## Regуларни изрази

- ▶ dosad izložene funkcije za operacije sa stringovima ne mogu rješiti širok skup problema koji se pojavljuju u realnom okruženju
- ▶ regularni izrazi (regular expressions) puno su fleksibilniji → omogućuju usporedbu stringa s **uzorkom** znakova
  - primjer: pronaći skup od tri uzastopna broja nakon kojih slijedi zarez ili točka, te iza kojih se nalazi najmanje jedno a najviše pet slova
- ▶ regularni izrazi se načelno mogu izraziti pomoću string-funkcija i petlje (ili dvije), no intuitivniji su i brži u odnosu na kombiniranje petlji i if-ova s string-funkcijama
- ▶ međutim, same string-funkcije su u pravilu brže od regularnih izraza, pa ako se problem može rješiti funkcijama, onda ih je i bolje primjeniti
- ▶ problem regularnih izraza je što naoko izgledaju poput grčkog pisma ☺

33

33

## Regуларни изрази – примјер

- ▶ Problem: kako provjeriti ispravnost domene?

```
$url = "https://www.vsite.hr/";  
if (preg_match('#^(http|https|ftp)://([A-Z0-9][A-Z0-9_-]*(?:.[A-Z0-9][A-Z0-9_-]*)+):?(d+)?/?#i', $url))  
{  
    echo „URL je ispravan.”;  
}  
else  
{  
    echo „Greška u URL-u.”;  
}
```

34

34

## Regуларни изрази

- ▶ regularni izrazi se stavljaju unutar delimitera, obično je to kosa crta / no može se upotrijebiti i poneki drugi znak (ipak ne bilo koji)
- ▶ preg\_match(uzorak, string); vraća 1 ako je pronašla poklapanje s uzorkom, odnosno 0 ako ga nije pronašla
- ▶ znak ^ u uzorku negira znakove koji mu slijede

```
echo preg_match("/p[aeiou]t/","Pronaći put..."); //ispisuje 1
echo preg_match("/p[aeiou]t/","Lupi petama"); //ispisuje 1
echo preg_match("/p[aeiou]t/","Netko je protiv"); //ispisuje 0
echo preg_match("/p[aeiou]t/","Leptir"); //ispisuje 0
echo preg_match("/p[aeiou]t/","Put"); //ispisuje 0
echo preg_match("/p[^aeiou]t/","Lupi petama"); //ispisuje 0
echo preg_match("/p[^aeiou]t/","Netko je protiv"); //ispisuje 0
```

35

35

## Regуларни изрази

- ▶ raspon vrijednosti definira se s znakom -
- ▶ alternativna vrijednost odjeljuje se znakom |

```
echo preg_match("/[0-9]/","vuk i 7 kozlića"); //ispisuje 1
echo preg_match("/[0123456789]/","vuk i 7 kozlića"); //ispisuje 1
echo preg_match("/p[a-z]\/","Plati pa idemo"); //ispisuje 1
echo preg_match("/P[a-zA-Z]\/","PARIZ"); //ispisuje 1
echo preg_match("/p[a-zA-Z]\/","Lab p5"); //ispisuje 0
echo preg_match("/magla | rosa/","Bila je magla jutros");
//ispisuje 1
echo preg_match("/magla | rosa/","Bila je rosa jutros");
//ispisuje 1
echo preg_match("/magla | rosa/","Bila je kiša jutros");
//ispisuje 0
```

36

36

# Regуларни изрази

- ▶ квантifikатори – говore колико пута поновити узорак

| квантifikатор | зnačenje                     |
|---------------|------------------------------|
| ?             | 0 или 1                      |
| *             | 0 или више                   |
| +             | 1 или више                   |
| {n}           | точно n пута                 |
| {n,m}         | најманje n, а највише m пута |
| {n,}          | најманje n пута              |

```
echo preg_match("/va+u/","vaaaaau"); //ispisuje 1  
echo preg_match("/va+u/","vuuu"); //ispisuje 0  
echo preg_match("/va?u/","vaaaaau"); //ispisuje 0  
echo preg_match("/va*u/","vu"); //ispisuje 1
```

37

37

# Regуларни изрази

- ▶ заhtijevanje tražene vrijednosti na početku stringa definira se s znakom ^ ispred uzorka
  - ▶ заhtijevanje tražene vrijednosti na kraju stringa definira se s znakom \$ iza uzorka
  - ▶ delimitери mogu biti i drugi znakovi osim /
- ```
echo preg_match("/^jog/","bavi se jogom"); //ispisuje 0  
echo preg_match("/^jog/","jogurt ujutro"); //ispisuje 1  
echo preg_match("/mir$/","ratimir"); //ispisuje 1  
echo preg_match("/mir$/","nemiran"); //ispisuje 0  
echo  
preg_match("#/?q=hr/node/#","http://www.vsite.hr/?q=hr/node/84");  
//ispisuje 1  
echo  
preg_match("/\/?q=hr\/*node\//","http://www.vsite.hr/?q=hr/node/84  
"); //ispisuje 1
```

38

38

## Regуларни изрази

- ▶ класе знакова – именовани скупови знакова које је могуће користити у узорку

klasa	знације
[:alnum:]	алфанимерички знакови
[:alpha:]	слови
[:digit:]	бројеви
[:lower:]	мало слово
[:upper:]	вељко слово
[:punct:]	интерпункцијски знак

39

39

## Regуларни изрази

```
echo preg_match("/PHP/","PHP је закон!");
echo preg_match("/PHP/","Што је PHP?");

echo preg_match("/^PHP/","PHP је закон!");
echo preg_match("/^PHP/","Што је PHP?");

echo preg_match("/PHP$/","Волим PHP");
echo preg_match("/PHP$/","Што је PHP?");

echo preg_match("/^PHP$/","Језик PHP је закон!");
echo preg_match("/^PHP$/","PHP");
```

40

40

## Regуларни изрази

```
echo preg_match("/[12345]/","1a");
echo preg_match("/[12345]/","76");
echo preg_match("/[^12345]/","1a");
echo preg_match("/^[12345]/","54");
```

41

41

## Regуларни изрази

```
echo preg_match("/bana?na/","banaana");
echo preg_match("/bana?na/","banna");
echo preg_match("/bana?na/","banana");

echo preg_match("/bana+na/","banana");
echo preg_match("/bana+na/","banna");
echo preg_match("/bana+na/","banaana");

echo preg_match("/bana*na/","banna");
echo preg_match("/bana*na/","banana");
echo preg_match("/bana*na/","banaaaana");
echo preg_match("/bana*na/","bnana");
```

42

42

## Regуларни изрази

- ▶ Функција preg\_replace() замјенjuje сва појављivanja traženog узорка с јеленим низом znakova
- ▶ Primjer: u tekstu zamijeniti uzastopna ponavljanja točke

```
$optimizacija=<<<SEM
```

Marketing tražilica (search engine marketing) je jedan od oblika Važno je biti prisutan na stranici s rezultatima pretraživanja, t rangiran... Korisnici uglavnom pregledavaju prvu stranicu s rezul Procjenjuje se da više od 80% cijelog prometa internetom stvaraju Najmanje 73% online kupaca turističkih usluga koristi tražilicu kupnjom....

Tražilice donose više od 50% posjetitelja web sjedištu.

```
SEM;
```

```
$optimizacija = preg_replace("/\.+/i", ".", $optimizacija);  
echo $optimizacija;
```

Marketing tražilica (search engine marketing) je jedan od oblika internetskog marketinga. Važno je biti prisutan rezultatima pretraživanja, te biti što više rangiran. Korisnici uglavnom pregledavaju prvu stranicu s rezultatim od 80% cijelog prometa internetom stvaraju tražilice. Najmanje 73% online kupaca turističkih usluga koristi rezultira kupnjom. Tražilice donose više od 50% posjetitelja web sjedištu.

43

## Regуларни изрази

- ▶ Primjer: isticanje јеленог скупа znakova

```
$sazetak = "Optimizacija za tražilice predstavlja strukturirani p  
kojima se želi postići bolja pozicija tvrtke na stranici s rezult  
Tražilice predstavljaju strukturirani pristup upotrebi tehnika kojima se želi postići bolja pozicija tvrtke na  
rezultatima pretraživanja. Tražilice tijekom vremena mijenjaju svoj algoritam kako bi napravile što bolje rangiranje  
rezultata, stoga je optimizacija web sjedišta kontinuirani proces, ali i izazov za stručnjake. Cilj ovog istraživanja je utvrđivanje  
strategije za optimizaciju web sjedišta hotela u Dalmaciji. Osim standardnih elemenata optimizacije na samoj stranici  
sještja, u kojoj mjeri su optimizirana web sjedišta hotela u Dalmaciji. Osim standardnih elemenata optimizacije na samoj stranici  
(oznaka naslova, metu opis, korištenje zaglavja, postojanje stranice 404 i korištenje analitičkog alata) istraživanjem je  
analizirano i brzina učitavanja stranice te optimiziranost za mobilne uređaje. Svega 6% hotela zadovoljilo je sve analizirane  
elemente optimizacije na samom web sjedištu, dok najveći udio, 70%, zadovoljava između tri i pet elemenata. Za vrednovanje  
optimizacije izvan web sjedišta korištene su dvije široko prihvaćene metrike (mozRank i Domain Authority), a za procjenu p  
SimilarWeb.";
```

```
$sazetak = preg_replace("/(optimiz)/i",  
'<span style="background:#5fc9f6">\1</span>', $sazetak);  
echo $sazetak;
```

44

## Regуларни изрази

- ▶ Ако је регуларни израз дугачак, тешко је разумљив
- ▶ Помоћ при провери визуализацијом поклапајућих узорака:  
<https://regex101.com/>

45

45



# P6: Objektno orijentirano programiranje u PHP-u

Kolegij: Programiranje u PHP  
Predavač: Dalibor Bužić

## Klase i objekti

- ▶ klasa je korisnički definiran tip podatka
  - sadrži podatke, funkcije (operacije za rukovajte podacima) te pristupna ograničenja nad funkcijama i podacima
  - funkcije unutar klase nazivamo i postupcima (metodama)
  - klasa je nacrt po kojem će se stvarati objekti
- ▶ objekt je instanca (primjer) klase
- ▶ odnos klase i objekta je poput odnosa tipa integer i konkretnie varijable \$x tog tipa
- ▶ iz jedne klase može nastati neograničen broj objekata

## Stvaranje klase

```
class automobil
{
    public $boja;
    public $GodProizvodnje;
    public $TrenutnaBrzina;
    function Ubrzaj()
    {
        $this->TrenutnaBrzina +=5;
        echo "Brzina auta je: $this->TrenutnaBrzina <br />";
    }
}
▶ $boja, $GodProizvodnje i $TrenutnaBrzina su podatkovni, a Ubrzaj() funkcijski članovi klase
▶ pomoću $this pristupamo podatkovnom članu unutar klase
```

3

3

## Stvaranje objekta

```
class automobil
{
    public $boja;
    public $GodProizvodnje;
    public $TrenutnaBrzina;
    function Ubrzaj()
    {
        $this->TrenutnaBrzina +=5;
        echo "Brzina auta je: $this->TrenutnaBrzina <br />";
    }
}
$auto1=new automobil();
$auto1->TrenutnaBrzina=40;
$auto1->Ubrzaj();
var_dump($auto1);
```

```
Brzina auta je: 45
object(automobil)#1 (3) { ["boja"]=> NULL ["GodProizvodnje"]=> NULL
["TrenutnaBrzina"]=> int(45) }
```

4

4

## Parametri postupka

- ▶ postupci mogu primati parametre:

```
class automobil
{
    public $boja;
    public $GodProizvodnje;
    public $TrenutnaBrzina;
    function Ubrzaj($koliko)
    {
        $this->TrenutnaBrzina +=$koliko;
        echo "Brzina auta je: $this->TrenutnaBrzina <br />";
    }
}
$auto1=new automobil();
$auto1->TrenutnaBrzina=40;
$auto1->Ubrzaj(25);
```

5

5

## Povratna vrijednost postupka

- ▶ postupci mogu vraćati parametre:

```
class automobil
{
    public $boja;
    public $GodProizvodnje;
    public $TrenutnaBrzina;
    function ImaGodina():int
    {
        $dob=date("Y") - $this->GodProizvodnje;
        return $dob;
    }
}
$auto1=new automobil();
$auto1->TrenutnaBrzina=40;
$auto1->GodProizvodnje=2012;
echo "Auto je star " . $auto1->ImaGodina() . " godina <br />";
```

6

6

# Čahurenje

- čahurenje (skrivanje podataka ili enkapsulacija)  
ostvaruje se pomoću modifikatora pristupa private

```
class automobil
{
    public $boja;
    public $GodProizvodnje;
    private $TrenutnaBrzina;
    function Ubrzaj($koliko)
    {
        $this->TrenutnaBrzina +=$koliko;
        echo "Brzina auta je: $this->TrenutnaBrzina <br />";
    }
}
$auto1=new automobil();
$auto1->TrenutnaBrzina=40;
$auto1->Ubrzaj(25); (!) Fatal error: Cannot access private property automobil::$TrenutnaBrzina in
```

- samo kod unutar klase može pristupiti podatkovnom članu deklariranom sa private!

7

```
class automobil
{
    public $boja;
    public $GodProizvodnje;
    private $TrenutnaBrzina;
    function __get($name) {
        return $this->$name;
    }
    function __set($name, $value) {
        $this->$name=$value;
    }
    function Ubrzaj()
    {
        $this->TrenutnaBrzina +=5;
    }
}
$auto1=new automobil();
$auto1->TrenutnaBrzina=40;
$auto1->Ubrzaj();
```

## \_\_set() i \_\_get()

8

8

## Kontrola vrijednosti

```
class automobil
{
    private $GodProizvodnje;
    function __get($name) {
        return $this->$name;
    }
    function __set($name, $value) {
        if (($name=="GodProizvodnje") && ($value>1950)) {
            $this->GodProizvodnje=$value;
        }
    }
    $auto1=new automobil();
    $auto1->GodProizvodnje=1928;
    var_dump($auto1->GodProizvodnje); //NULL
```

9

9

## Konstruktor

```
class automobil
{
    public $boja;
    public $GodProizvodnje;
    public $TrenutnaBrzina;
    function __construct() {
        $this->TrenutnaBrzina=40;
    }
    function Ubrzaj($koliko)
    {
        $this->TrenutnaBrzina +=$koliko;
    }
    $auto1=new automobil();
    echo "$auto1->TrenutnaBrzina <br />";
    $auto1->Ubrzaj(25);
    echo $auto1->TrenutnaBrzina;
```

10

10

## Konstruktor

```
class automobil
{
    public $boja;
    public $GodProizvodnje;
    public $TrenutnaBrzina;
    function __construct($brz) {
        $this->TrenutnaBrzina=$brz;
    }
}
$auto1=new automobil(20);
$auto2=new automobil(5);
$auto3=new automobil();

echo "auto1 vozi: $auto1->TrenutnaBrzina      km/h<br />";
echo "auto2 vozi: $auto2->TrenutnaBrzina      km/h<br />";
echo "auto3 vozi: $auto3->TrenutnaBrzina      km/h<br />";
```

11

11

## Konstruktor

```
class automobil
{
    public $boja;
    public $GodProizvodnje;
    public $TrenutnaBrzina;
    function __construct($brz=0) {
        $this->TrenutnaBrzina=$brz;
    }
}
$auto1=new automobil(20);
$auto2=new automobil(5);
$auto3=new automobil();

echo "auto1 vozi: $auto1->TrenutnaBrzina      km/h<br />";
echo "auto2 vozi: $auto2->TrenutnaBrzina      km/h<br />";
echo "auto3 vozi: $auto3->TrenutnaBrzina      km/h<br />";
```

12

12

## Destruktor

```
class automobil
{
    public $boja;
    public $GodProizvodnje;
    public $TrenutnaBrzina;
    function __construct($brz) {
        $this->TrenutnaBrzina=$brz;
    }
    function __destruct() {
        echo "uništavam objekt!!! <br />";
    }
}
$auto1=new automobil(20);
$auto2=new automobil(5);
echo "auto1 vozi: $auto1->TrenutnaBrzina    km/h<br />";
echo "auto2 vozi: $auto2->TrenutnaBrzina    km/h<br />";
```

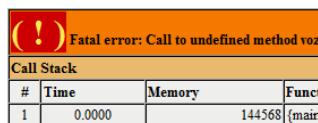
auto1 vozi: 20 km/h  
auto2 vozi: 5 km/h  
uništavam objekt!!!  
uništavam objekt!!!

13

13

## Nasljeđivanje

```
class vozilo
{
    public $boja;
    public function f1()
    {
        echo "Funkcija roditeljske klase <br />"; }
}
class automobil extends vozilo
{
    public $TrenutnaBrzina;
    function f2()
    {
        echo "Funkcija klase djeteta <br />";}
}
$vozilo1 =new vozilo();
$auto1=new automobil();
$vozilo1->f1();
$vozilo1->f2(); ←
$auto1->f1();
$auto1->f2();
```



14

14

```

class vozilo          public, private, protected
{
    public function f1()
    {   echo "Funkcija public roditeljske klase <br />"; }
    protected function f2()
    {   echo "Funkcija protected roditeljske klase <br />";}
    private function f3()
    {   echo "Funkcija private roditeljske klase <br />"; }
}
class automobil extends vozilo
{
    function __construct()
    { $this->f1();
      $this->f2();
      $this->f3(); } }

$auto1=new automobil();
$auto1->f1();
$auto1->f2(); ←
$auto1->f3(); ←

```

Funkcija public roditeljske klase  
Funkcija protected roditeljske klase

( ! ) Fatal error: Call to private method vozilo::f3() from context...  
Call Stack

#	Time	Memory	Function
1	0.0030	144736	{main}()
2	0.0030	145072	automobil->__construct()

( ! ) Fatal error: Call to protected method vozilo::f2() from context...  
Call Stack

#	Time	Memory	Function
1	0.0010	145264	{main}()

15

15

```

class vozilo          Nadjačavanje (overriding)
{
    public $v="A";
    public function f1()
    { echo "Vrijednost u klasi vozilo je $this->v <br />";}
}
class automobil extends vozilo
{
    public $v="B";
    public function f1()
    { echo "Vrijednost u klasi automobil je $this->v <br />";}
}
$vozilo1 = new vozilo();
$auto1=new automobil();
$vozilo1->f1();
$auto1->f1();

```

Vrijednost u klasi vozilo je A  
Vrijednost u klasi automobil je B

16

16

```
class vozilo    Nadjačavanje (overriding)
{
    public $v="A";
    final public function f1()
    { echo "Vrijednost u klasi vozilo je $this->v <br />";}
}
class automobil extends vozilo
{
    public $v="B";
    public function f1()
    { echo "Vrijednost u klasi automobil je $this->v <br />"; }
}
$vozilo = new vozilo();
$auto1=new automobil();
$vozilo->f1();
$auto1->f1();
```

( ! ) Fatal error: Cannot override final method vozilo:

17

17



## P7: PHP i web forme

Kolegij: Programiranje u PHP  
Predavač: Dalibor Bužić

1

## HTTP

- ▶ HTTP (HyperText Transfer Protocol) prenosa informacije na Webu
- ▶ HTTP upravlja načinima kako web pretraživač zahtjeva datoteke od web poslužitelja i kako ih server vraća pretraživaču

GET /index.html HTTP/1.1

Mozilla/5.0 (Windows NT 6.1; rv:25.0) Gecko/20100101  
Firefox/25.0  
Accept: image/gif, text/\*, \*/\*

---

HTTP/1.1 200 OK  
Date: Fri, 06 Nov 2013 09:33:12 GMT  
Server: Apache  
Content-Length: 2304  
Content-Type: text/plain

2

2

# Superglobalne varijable-nizovi

- ▶ PHP stvara šest varijabli – asocijativnih nizova –koje sadrže korisne parametre i informacije:
  - \$\_GET – sadrži parametre GET zahtjeva (GET podaci iz obrasca)
  - \$\_POST – sadrži parametre POST zahtjeva (POST podaci iz obrasca)
  - \$\_FILES – sadrži informacije o datotekama poslanih na poslužitelj
  - \$\_COOKIE – sadrži vrijednosti kolačića
  - \$\_SERVER – sadrži informacije o poslužitelju
  - \$\_ENV – sadrži informacije o okolini izvođenja (varijable okoline)
  - \$\_SESSION – sadrži informacije o varijablama sjednice
  - \$\_GLOBALS – sadrži sve globalne varijable
- ▶ ove varijable su superglobalne – vidljive su su svim dijelovima programskog koda (dakle, i unutar funkcija)

3

3

## \$ \_ SERVER

```
echo $_SERVER['PHP_SELF'] . "<br>";  
echo $_SERVER['SERVER_SOFTWARE'] . "<br>";  
echo $_SERVER['SERVER_NAME'] . "<br>";  
echo $_SERVER['GATEWAY_INTERFACE'] . "<br>";  
echo $_SERVER['SERVER_PROTOCOL'] . "<br>";  
echo $_SERVER['SERVER_PORT'] . "<br>";  
echo $_SERVER['REQUEST_METHOD'] . "<br>";
```

```
/uvodforme/index.php  
Apache/2.4.4 (Win32) PHP/5.4.16  
localhost  
CGI/1.1  
HTTP/1.1  
80  
GET
```

4

4

# Web obrasci

- ▶ web obrasci se definiraju unutar `<form>` oznake
- ▶ obrasci uobičajeno sadržavaju polja za unos teksta, gumbe za odabir (radio buttons), kućice za potvrđivanje (check box), padajuće liste (list box), polja za odabir datoteke i gume (button)
- ▶ u jednoj HTML stranici može biti više obrazaca – više `<form>` oznaka s pripadnim elementima

```
<form method="get" action="UnesiPodatke.php"
      enctype="text/plain">
    <!-- elementi obrasca -->
</form>
```

- ▶ *method* – način slanja podataka
- ▶ *action* – naziv skripte koja će obraditi podatke
- ▶ *enctype* – način kodiranja poslanih podataka

5

5

## Elementi web obrasca

- ▶ kućica za tekst

```
<input type="text" name="imeKorisnika"
       maxlength="30" size="20" />
```

- ▶ kućica za lozinku

```
<input type="password" name="lozinka"
       maxlength="30" size="20" />
```

- ▶ višelinjski tekst

```
<textarea name="opisProblema" cols="50" rows="4">
</textarea>
```

- ▶ kućica za potvrđivanje

```
<input type="checkbox" name="artiklAktivan"
       checked="checked" />
```

6

6

# Elementi web obrasca

## ► gumb za odabir

```
<input type="radio" name="vrstaStudiranja" value="redovni"  
checked="checked" />  
<input type="radio" name="vrstaStudiranja"  
value="izvanredni" />
```

## ► Lista za odabir

```
<select name="vrstaStudiranja">  
  <option value="redovni" selected="selected">  
    Redovni student  
  </option>  
  <option value="izvanredni">  
    Izvanredni student  
  </option>  
</select>
```

7

7

# Elementi web obrasca

## ► polje za odabir datoteke

```
<input type="file" name="datoteka" size="40" />
```

## ► skriveno polje

```
<input type="hidden" name="poljeZaPodatke" />
```

## ► gumbi

- <input type="submit" value="Pošalji" />
- <input type="reset" value="Obriši" />
- <input type="button" value="Povratak" />

## ► ...

8

8

## GET i POST postupci

- ▶ klijent može serveru proslijediti podatke s forme pomoću GET i POST postupaka
- ▶ **method** atribut **form** oznake (taga) određuje koji postupak koristi određena forma
- ▶ GET zahtjev ugrađuje parametre forme u sam URL – počinje s znakom ? i naziva se upitni (eng. query) string
- ▶ POST zahtjev proslijeđuje parametre forme tako što ih ugradi u tijelo HTTP zahtjeva, i on ne mijenja URL
- ▶ web pretraživači mogu privremeno pohraniti (cache) stranice koje predstavljaju odgovor na GET zahtjev, a odgovore na POST zahtjev ne mogu

9

9

## GET i POST postupci

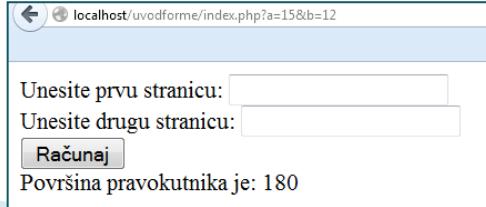
- ▶ i `$_POST` i `$_GET` i `$_FILES` su asocijativni nizovi čiji ključevi su imena parametara, a vrijednosti su vrijednosti tih parametara na formi
- ▶ ako naziv polja na formi sadrži točku, ona se automatski pretvara u podvlaku, budući da sintaksa PHP-a ne dopušta točku kao legalan znak u identifikatoru
  - zato programer u PHP kodu koristi podvlaku

10

10

## GET postupak

```
<form action=<?php echo $_SERVER['PHP_SELF']; ?>" method="GET">
Unesite prvu stranicu: <input type="text" name="a" /><br />
Unesite drugu stranicu: <input type="text" name="b" /><br />
    <input type="submit" value="Računaj"/>
</form>
<?php
if (!empty($_GET['a']) && (!empty($_GET['b']))) {
    $strA=$_GET['a'];
    $strB=$_GET['b'];
    echo "Površina pravokutnika je: " . $strA*$strB;
}
?>
```



11

11

## POST postupak

```
<form action=<?php echo $_SERVER['PHP_SELF']; ?>" method="POST">
Unesite prvu stranicu: <input type="text" name="a" /><br />
Unesite drugu stranicu: <input type="text" name="str.b" /><br />
    <input type="submit" value="Računaj"/>
</form>
<?php
if (!empty($_POST['a']) && (!empty($_POST['str.b'])))) {
    $strA=$_POST['a'];
    $strB=$_POST['str.b'];
    echo "Površina pravokutnika je: " . $strA*$strB;
}
?>
```

12

12

## Prikaz rezultata u istoj formi (sticky form)

- da bi se rezultat upita prikazao na formi na kojoj je upit i napravljen, može se unesena vrijednost na formi iskoristiti kao podrazumijevana vrijednost u dijelu HTML-a koji stvara polje za unos

```
<?php if(!isset($_GET['posalji'])){ $strA=''; $strB='';}  
    else{ $strA=$_GET['a']; $strB=$_GET['b'];}  
?  
<form action=<?php echo $_SERVER['PHP_SELF']; ?>" method="GET">  
Unesite 1. stranicu: <input type="text" name="a"  
    value=<?php echo $strA;?>>/><br />  
Unesite 2. stranicu: <input type="text" name="b"  
    value=<?php echo $strB;?>>/><br />  
<input type="submit" name="posalji" value="Računaj" />  
Površina pravokutnika je: 180  
</form>  
<?php  
if (!empty($_GET['a']) && (!empty($_GET['b']))) {  
    $strA=$_GET['a']; $strB=$_GET['b'];  
    echo "Površina pravokutnika je: " . $strA*$strB;  
}  
?>
```

13

## Višestruki odabir

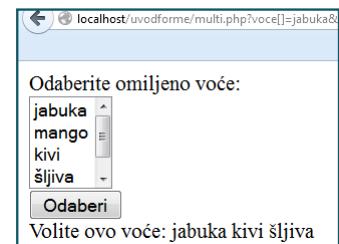
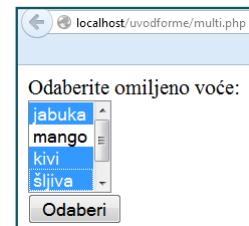
- HTML liste za odabir koje se stvaraju pomoću select oznake omogućuju višestruki odabir
- kako bi PHP prepoznao, ime polja u HTML formi mora završavati s []
- umjesto liste za odabir mogu se koristiti i kućice za potvrđivanje (checkbox)
- u sljedeća dva primjera samo se HTML mijenja; PHP kod je isti!

14

14

## Višestruki odabir – lista

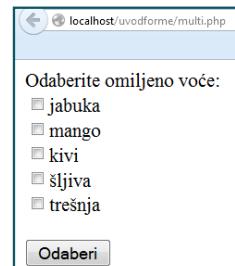
```
<form action=" <?php echo $_SERVER['PHP_SELF']; ?>" method="GET">
Odaberite omiljeno voće:<br />
<select name="voce[]" multiple>
    <option value="jabuka">jabuka</option>
    <option value="mango">mango</option>
    <option value="kivi">kivi</option>
    <option value="šljiva">šljiva</option>
    <option value="trešnja">trešnja</option>
</select><br />
<input type="submit" name="unesi" value="Odaberis" />
</form>
<?php
if (array_key_exists('unesi', $_GET)) {
    $finoVoce=join(' ', $_GET['voce']);
    echo "Volite ovo voće: $finoVoce ";
}
?>
```



15

## Višestruki odabir – kućice za potvrđivanje

```
<form action=" <?php echo $_SERVER['PHP_SELF']; ?>" method="GET">
Odaberite omiljeno voće:<br />
    <input type="checkbox" name="voce[]" value="jabuka">jabuka<br />
    <input type="checkbox" name="voce[]" value="mango">mango<br />
    <input type="checkbox" name="voce[]" value="kivi">kivi<br />
    <input type="checkbox" name="voce[]" value="šljiva">šljiva<br />
    <input type="checkbox" name="voce[]" value="trešnja">trešnja<br />
<br />
<input type="submit" name="unesi" value="Odaberis" />
</form>
<?php
if (array_key_exists('unesi', $_GET)) {
    $finoVoce=join(' ', $_GET['voce']);
    echo "Volite ovo voće: $finoVoce ";
}
?>
```



16

## Problem s varijablim \$\_SERVER["PHP\_SELF"]

- ▶ Ako na stranici testna\_forma.php imamo sljedeću formu:

```
<form method="post" action="<?php echo  
$_SERVER["PHP_SELF"];?>">
```

- ▶ sve je u redu ako korisnik u adresnu traku unese normalni URL kao npr. [http://www.domena.com/testna\\_forma.php](http://www.domena.com/testna_forma.php)

jer će se kod interpretirati kao:

```
<form method="post"  
action="testna_forma.php">
```

No ako korisnik u adresnu traku unese ovo:

```
http://www.domena.com/testna_forma.php/%22%3E%3Cscri  
pt%3Ealert('hakirano')%3C/script%3E
```

17

17

## Problem s varijablim \$\_SERVER["PHP\_SELF"]

- ▶ Kod s prethodnog slajda će se interpretirati kao:

```
<form method="post"  
action="testna_forma.php/"><script>alert(  
'hakirano')</script>
```

- ▶ Ovaj kod dodaje skriptnu oznaku i alert naredbu. Kad se stranica učita, JavaScript kod će se izvršiti (korisnik će vidjeti alert box)

- Primjer je bezopasan, no ukazuje kako se može iskoristiti varijabla PHP\_SELF

18

18

## Problem s varijablim \$\_SERVER["PHP\_SELF"]

### ► Rješenje:

- Korištenje funkcije htmlspecialchars() koja specijalne znakove pretvara u HTML entitete

```
<form method="post" action="<?php echo  
htmlspecialchars($_SERVER["PHP_SELF"]);?>">
```

- evaluira se kao:

```
<form method="post"  
action="testna_forma.php/&quot;&lt;script&gt;a  
lert('hacked')&lt;/script&gt;">
```

- Pokušaj zloupotrebe propada, nema štete

19

19

## Održavanje stanja informacija

- kad web poslužitelj izvrši zahtjev klijenta za web stranicom, veza između klijenta i poslužitelja se prekida
- kako poslužitelj može prepoznati niz zahtjeva koji dolaze od istog klijenta?
  - košarica kupca u web trgovini – stavljanje artikla u košaricu, uklanjanje, što je u košarici kad želi završiti kupnju tj. platiti?
- praćenje sjednica (session tracking):
  - skrivena polja na formi za prijenos informacija
  - promjena URL-ova (URL rewriting)
  - kolacići (cookies)
  - ugrađeni PHP-ov sustav praćenja sjednica (session-tracking system)

20

20

## Skrivena polja na formi

- ▶ dva načina:
  - skrivena polja na formi za prijenos informacija → svako polje prenosi informacije o naručenom artiklu
  - skrivena polja na formi za prijenos informacija → svaki korisnik dobiva jedinstveni identifikator pa se taj ID prenosi pomoću samo jednog skrivenog polja forme
- ▶ ova tehnika funkcioniра na svim internetskim preglednicima
- ▶ nedostaci:
  - radi samo s dinamički generiranim formama
  - spor način pohranjivanja velikih količina podataka
  - poprilično nesigurno
- ▶ općenito nije korisna poput drugih tehnika

21

21

## Promjena URL-ova

- ▶ svaki URL na koji korisnik može kliknuti dinamički se mijenja → ugrađuju se dodatne informacije
- ▶ primjerice ako se svakom pojedinom korisniku dodjeljuje jedinstveni ID, on se može uključiti u URL
  - <http://www.ponuda-caja.com/katalog.php?userid=918>
- ▶ ova tehnika radi sa svim dinamički generiranim dokumentima (ne samo s formama)
- ▶ promjena URL-ova može biti vremenski zahtjevna

22

22

## Kolačići

- ▶ najraširenija tehnika pamćenja stanja informacija
- ▶ kolačić je mali skup informacija koje server može dati klijentu
- ▶ pri svakom sljedećem zahtjevu, klijent vraća tu informaciju natrag serveru, čime se identificira
- ▶ kolačići su korisni i kod ponovnih posjeta web stranici (npr nekoliko dana kasnije)
- ▶ nedostatak ove tehnike je što korisnici mogu isključiti kolačice u web pregledniku

23

23

## Kolačići

- ▶ za slanje kolačića pregledniku koristi se funkcija
  - `setcookie(name [, value [, expire [, path [, domain [, secure]]]]);`
  - name je jedinstveno ime kolačića
  - value je vrijednost
  - expire označava datum do kojeg kolačić postoji
  - path određuje da će preglednik vratiti kolačić samo s URL-a u ovoj putanji
  - domain označava da preglednik vraća kolačić samo s URL-ova na ovoj domeni
  - secure označava da li prenosi kolačić putem https veze (podrazumijevano je false)

24

24

## Kolačići

- ▶ pristup kolačiću:

```
echo $_COOKIE["brojPregleda"];  
▶ novostvoreni kolačić nije dostupan skripti preko  
$_COOKIE dokle god se ne napravi sljedeći zahtjev  
preglednika  
    ◦ prilikom prvog pokretanja skripte, skripta šalje kolačić  
        pregledniku, a preglednik ne vraća kolačić poslužitelju odmah,  
        već pri prvom sljedećem zahtjevu
```

```
setcookie("brojPregleda", 5,0,"/", "",false, true);  
if(isset($_COOKIE["brojPregleda"]))  
    echo "Stranice pregledane ". $_COOKIE["brojPregleda"]. "  
puta";  
else {  
    echo "Podatak ne postoji"; //ispisuje se ova poruka  
}
```

- ▶ analogno vrijedi i za ažuriranje vrijednosti kolačića –  
\$\_COOKIE sadrži stare vrijednosti tijekom izvršavanja  
skripte; tek po ponovnom pokretanju skripte (reload  
stranice u pregledniku) ažurira se vrijednost

25

25

## Sjednice

- ▶ PHP ima ugrađenu podršku za sjednice, i obavlja  
svu manipulaciju kolačićima umjesto programera  
kako bi osigurao postojane varijable dostupne u  
različitim stranicama te u višestrukim posjetima  
web sjedištu
- ▶ sjednice omogućuju jednostavno stvaranje  
višestraničnih formi (košarice u internetskom  
dučanu), spremanje autentifikacijskih podataka  
korisnika od stranice do stranice te pohranu  
korisničkih postavki
- ▶ kod prvog posjeta korisniku se izdaje jedinstveni  
ID sjednice (uobičajeno u kolačiću PHPSESSID)
- ▶ ako preglednik ne dozvoljava kolačice, ID  
sjednice se ugrađuje u URL

26

26

## Sjednice

- ▶ svaka sjednica ima svoje pohranjene podatke
- ▶ kad stranica 'počinje' mogu se registrirati varijable koje vrijednosti učitavaju iz pohranjenih podataka
- ▶ kad stranica 'završava', vrijednosti se mogu pohraniti
- ▶ registrirane varijable postoje između stranica, a promjene varijable na jednoj stranici vidljive su na drugoj stranici
- ▶ sjednica započinje funkcijom **session\_start()** koja učitava registrirane varijable u asocijativni niz **\$\_SESSION** (ključevi niza su imena varijabli)
- ▶ za kraj sjednice potrebno je pozvati **session\_destroy()** funkciju koja uklanja podatke tekuće sjednice, ali ne uklanja kolačiće
- ▶ zato kad se korisnik ponovno vrati na stranicu, imat će isti ID kao i zadnji put, jedino neće biti podataka

27

27

## Sjednice

```
session_start();  
echo session_id();  
//ispisuje npr. 4rbq6qcbsj0eubu6se1o5hfv31
```

- ▶ varijablama sjednice pristupa se preko asocijativnog niza **\$\_SESSION**; stvaranje nove varijable je zapravo dodavanje novog elementa u niz:

```
$SESSION["korisnik"]=$imeKorisnika;  
čitanje vrijednosti varijable sjednice je pristupanje članu asocijativnog niza $SESSION  
if(isset($SESSION["korisnik"]))  
echo $SESSION["korisnik"];
```
- ▶ varijable sjednice čuvaju se na serveru  
varijablama se pristupa preko identifikatora sjednice

28

28

```

<?php
    session_start(); // započinjanje sjednice
?>
<html>
    <head></head>
    <body>
        <?php
            if (!isset($_SESSION['ime']) && !isset($_POST['ime']))
                { //ako podataka nema, prikazuje se forma
                ?>
                    <form action=<?php echo $_SERVER['PHP_SELF']?> method="post">
                        <input type="text" name="ime">
                        <input type="submit" name="submit" value="Unesi ime"></form>
        <?php
    }
    else
        if (!isset($_SESSION['ime']) && isset($_POST['ime']))
            { // ako sjednica ne postoji, a forma je poslana
                // provjerava se jesu li proslijedeni svi podaci iz forme
            if (!empty($_POST['ime'])) {$_SESSION['ime'] = $_POST['ime'];
            $_SESSION['pocetak'] = time();
            echo "Dobro došao, " . $_POST['ime'] . " Stvorena je nova sjednica <br>";
            echo "Klikni <a href=" . $_SERVER['PHP_SELF'];
            echo ">ovdje</a> za osvježavanje stranice.<br>"}
        }

```

29

29

```

        // provjerava se jesu li proslijedeni svi podaci iz forme
if (!empty($_POST['ime'])) {$_SESSION['ime'] = $_POST['ime'];
$_SESSION['pocetak'] = time();
echo "Dobro došao, " . $_POST['ime'] . ". Stvorena je nova sjednica <br>";
echo "Klikni <a href=" . $_SERVER['PHP_SELF'];
echo ">ovdje</a> za osvježavanje stranice.<br>"}
else
{
    echo "GREŠKA: Nije uneseno ime!<br>";
}
else if (isset($_SESSION['ime'])) {
// ako sjednica postoji izračunaj vrijeme od početka sjednicce
echo "Evo nas opet, " . $_SESSION['ime'] ;
echo ". Ova sjednica je započeta prije ";
echo round((time() - $_SESSION['pocetak']) ) . " sekundi. <br>";
echo "Klikni <a href=". $_SERVER['PHP_SELF'];
echo ">ovdje</a> za osvježavanje stranice.<br>"}
}
?>
</body>
</html>

```

30

30



## P8: Pristup bazi podataka

Kolegij: Programiranje u PHP  
Predavač: Dalibor Bužić

### Pristup bazama podataka pomoću PHP-a

- ▶ PHP podržava rad s više od 20 SUBP-a, što uključuje najpopularnije komercijalne i open-source sustave
  - MySQL, PostgreSQL, Oracle, SQLite, MongoDB, MS SQL Server...
- ▶ dva načina pristupa bazama iz PHP-a:
  1. pomoću ekstenzija specifičnih za pojedinu bazu
    - brži, ali u slučaju migracije na drugu bazu (odnosno SUBP) potrebno mijenjati popriličnu količinu koda
  2. pomoću PDO-a (PHP Data Objects)
    - sloj apstrakcije koji skriva funkcije specifične za konkretni SUBP pa je migracija gotovo trivijalna
    - sporiji način

## Pristup bazama podataka pomoću PHP-a

- ▶ ipak, PDO ne obavlja pretvorbu SQL upita, o tome mora voditi računa programer
- ▶ Relacijska baza podataka (Oracle, MySQL...) sadrži tablice u kojima se u nalaze podaci promatrane problemske domene
- ▶ PHP s relacijskim bazama podataka komunicira pomoću SQL-a (Structured Query Language)

3

3

## Apache, MySQL i PHP

- ▶ Apache, MySQL i PHP su open-source projekti koji se mogu instalirati na velikom broju platformi
- ▶ najpopularniji su na Linuxu ("LAMP"), no sve veću popularnost dobivaju i na Windowsima
- ▶ MySQL je najpopularniji open-source SUBP
- ▶ PHP najčešće koristi upravo MySQL platformu
- ▶ AMP kombinacija je najpopularnija iz vrlo jednostavnog razloga: skladno funkcioniraju

4

4

## MySQL

- ▶ na jednom sustavu za upravljanje bazama podataka može se nalaziti više baza podataka
- ▶ jedna baza se uobičajeno sastoji od većeg broja tablica i drugih objekata
- ▶ pojedina tablica ima definirani broj stupaca i proizvoljan broj redova
- ▶ kod definicije stupaca nužno je odrediti kojeg tipa podataka će podaci u njemu biti
- ▶ osim tipa podataka mogu se definirati i ograničenja nad stupcem
- ▶ svaka tablica mora imati primarni ključ
- ▶ tablice u bazi moraju biti normalizirane

5

## MySQL – neki tipovi podataka

Tip	opis
CHAR(n)	niz duljine n znakova
VARCHAR(n)	niz duljine najviše n znakova
TEXT	najviše 65536 znakova
INT	cijeli broj u rasponu od -2147483648 do 2147483647
INT UNSIGNED	cijeli broj od 0 do 4294967295
FLOAT	decimalni brojevi
DOUBLE	decimalni brojevi
DECIMAL (z,d)	decimalni brojevi s ukupno z znamenki od čega je d decimalnih mesta

6

6

## MySQL – neki tipovi podataka

Tip	opis
DATE	datum u formatu YYYY-MM-DD
TIME	vrijeme u formatu hh:mm:ss
DATETIME	datum i vrijeme u formatu YYYY-MM-DD hh:mm:ss
BLOB	za spremanje binarnih podataka (datoteka)

7

7

## SQL

- ▶ SQL ima dvije skupine naredbi:
  - Data Definition Language (DDL) – naredbe CREATE, ALTER i DROP kojima se mijenja struktura baze podataka
    - sintaksa ovih naredbi nije standardizirana u mjeri u kojoj je to slučaj s drugom skupinom naredbi
  - Data Manipulation Language (DML) – naredbe za rad s podacima
    - INSERT, UPDATE, DELETE za unos promjenu i brisanje podataka
    - SELECT za dohvaćanje podataka iz baze → izvođenje

8

8

## SQL DDL

```
CREATE TABLE Dobavljac (
    dobavljacID INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
    nazivDob VARCHAR(60)          NOT NULL,
    adresa VARCHAR(70)            NOT NULL,
    telefon VARCHAR(20)           NOT NULL,
    PRIMARY KEY (dobavljacID)
);
```

9

9

## SQL – naredba SELECT

```
SELECT imena_stupca
FROM imena_tablica | logička_veza
[WHERE uvjeti]
[GROUP BY imena_stupaca_grupiranja]
[HAVING uvjeti]
[ORDER BY stupci_sortiranja]
```

10

10

## SQL – naredbe INSERT, UPDATE i DELETE

- ▶ dodavanje retka u tablicu:

```
INSERT INTO tablica (stupci)  
VALUES (vrijednosti)
```

```
INSERT INTO Dobavljac (dobavljacID,nazivDob,  
adresa, telefon) VALUES  
(1,"Kraš", "Ravnice 48, Zagreb", "01 2396 666");
```

promjena vrijednosti u retku

```
UPDATE tablica SET stupac=nova_vrijednost [...]  
WHERE uvjet
```

```
UPDATE Dobavljac SET telefon = "01 2396 111"  
WHERE dobavljacID = 1
```

11

11

## SQL – naredbe INSERT, UPDATE i DELETE

- ▶ brisanje retka iz tablice:

```
DELETE FROM tablica  
WHERE uvjet
```

```
DELETE FROM Dobavljac  
WHERE dobavljacID = 666
```

- ▶ kod naredbi UPDATE i DELETE treba biti oprezan što se stavlja u uvjet!

12

12

## Spajanje na MySQL

- ▶ Dva načina:
  - PDO – PHP Dana Objects
  - MySQLi Extension
- ▶ PDO može raditi s više od 10 različitih baza
- ▶ MySQLi radi samo s MySQL bazom
- ▶ Migracija projekta na drugu bazu značajno je jednostavnija ako se koristi PDO (mijenja se connection string i svega nekoliko upita)

13

13

## Primjer spajanja na MySQL i stvaranja baze podataka

- ▶ PDO – veza između PHP-a i baze podataka
  - Stvaranjem objekta iz ove klase uspostavlja se veza

```
<?php
try {
    $db= new PDO('mysql:host=localhost','root','');
    $db->exec('CREATE DATABASE IF NOT EXISTS trgovina
                CHARACTER SET utf8 COLLATE utf8_croatian_ci');
} catch (PDOException $e) {
    echo 'Greška pri spajanju s poslužiteljem: ' . $e-
>getMessage();
}
?>
```

- ▶ Veza se automatski zatvara završetkom izvođenja skripte
  - Za raniji izlazak naredba je `$db= null;`

14

14

## Važni postupci PDO-a

- ▶ lastInsertId() – vraća zadnji ubačeni ID
- ▶ prepare(\$sql) – vraća instancu PDOStatement za zadani upit
- ▶ query(\$sql) – izvodi upit i vraća PDOStatement
  - Za naredbu SELECT
- ▶ exec(\$sql) – izvodi upit i vraća broj redaka na koje je upit utjecao
  - Za naredbe INSERT, UPDATE i DELETE
- ▶ setAttribute(\$atribut, \$vrijednost) – mijenja postavke

15

15

## Stvaranje tablica

```
<?php  
$stvoridobavljaca='CREATE TABLE dobavljac (  
    dobavljacID INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
    nazivDob VARCHAR(60)          NOT NULL,  
    adresa VARCHAR(70)           NOT NULL,  
    telefon VARCHAR(20)          NOT NULL,  
    PRIMARY KEY (dobavljacID)  
)  
ENGINE = MyISAM';  
  
$stvorikategoriju='CREATE TABLE kategorija (  
    kategorijaID INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
    nazivKat VARCHAR(30)          NOT NULL,  
    PRIMARY KEY (kategorijaID)  
)  
ENGINE = MyISAM';
```

16

16

## Stvaranje tablica (...nastavak)

```
$stvoriProizvod='CREATE TABLE proizvod (
    proizvodID INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
    nazivPro VARCHAR(40)          NOT NULL,
    cijena DECIMAL(7,2)           NOT NULL,
    kolicina SMALLINT             NOT NULL DEFAULT 0,
    PRIMARY KEY (proizvodID) ) ENGINE = MyISAM';

try {
    $db= new
PDO('mysql:host=localhost;dbname=trgovina','root','');
    $db->exec("SET NAMES utf8");
    $db->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
    $db->exec($stvoriDobavljaca);
    $db->exec($stvoriKategoriju);
    $db->exec($stvoriProizvod);
} catch (PDOException $e) {
    echo 'Greška u stvaranju tablice: ' . $e->getMessage();
}
```

17

17

## Rad s podacima u tablici – PDOStatement

- ▶ upite možemo raditi instancom PDOStatement-a koja se dobije metodom `prepare($sql)`

```
$upit = $db->prepare('INSERT INTO Dobavljac
(dobavljacID,nazivDob, adresa, telefon) VALUES
(1,"Kraš", "Ravnice 48, Zagreb","01 2396 111"),
(2,"Labud", "Radnička cesta 173 r, Zagreb","01 2396 111"),
(3,"Podravka", "Ante Starčevića 32, Koprivnica",
"048 651 144")');

$upit->execute();
```

18

18

## Prepared Statement

- ▶ Pripremljena naredba (engl. *Prepared statement*) predstavlja mogućnost visoko efikasnog uzastopnog izvršavanja istih (ili sličnih) naredbi SQL-a
- ▶ Princip je sljedeći:
  1. Priprema – stvara se predložak naredbe SQL-a te se šalje bazi. Neke vrijednosti su nespecificirane, zovemo ih parametri, a predstavljeni su znakom upitnika ("?"). Npr:  
`INSERT INTO kategorija VALUES (?, ?)`
  2. SUBP parsira, predovi i izvršava optimizaciju predloška upita te pohranjuje rezultat bez izvršavanja
  3. Kasnije aplikacija povezuje vrijednosti s parametrima te SUBP sad izvršava naredbu. Naredbu se može izvršiti neograničen broj puta s različitim parametrima

19

## Prepared Statement – prednosti

- ▶ Prednosti korištenja pripremljenih naredbi u odnosu na izravno izvršavanje naredbi SQL-a su:
  - Pripremljene naredbe smanjuju vrijeme parsiranja jer se priprema upita obavlja samo jednom
  - Povezani parametri smanjuju prijenos mrežom budući da se svaki put šalju samo parametri, a ne cijeli upiti
  - Pripremljene naredbe korisne su u borbi protiv SQL injection napada jer se vrijednosti šalju kasnije i ne mora se raditi *escape*

20

## Rad s podacima u tablici – PDOStatement

- ▶ PDOStatement radi s rezerviranim mjestima  
(engl. *placeholders*)
  - imenovani – :naziv
  - neimenovani – ?

```
$upit= $db->prepare("INSERT INTO kategorija (kategorijaID, nazivKat) "
    . "VALUES (:kategorijaID, :nazivKat)");
$upit->execute(array(':kategorijaID'=>1, ':nazivKat'=>'juha'));
$upit->execute(array(':kategorijaID'=>2, ':nazivKat'=>'dodatak jelu'));


$upit= $db->prepare("INSERT INTO kategorija (kategorijaID, nazivKat) "
    . "VALUES (?, ?)");
$upit->execute(array(3,'čokolada'));
$upit->execute(array(4,'keksi'));
$upit->execute(array(5,'deterdžent'));
```

21

## Povezivanje parametara

- ▶ olakšano postavljanje upita
  - bindValue(\$stupac, \$varijabla)
    - rezerviranom mjestu pridružuje vrijednost varijable
  - bindParam(\$stupac, \$varijabla)
    - rezerviranom mjestu pridružuje ime varijable
  - stupac je naziv stupca ili redni broj rezerviranog mesta
    - rezervirana mjesta se broje od 1 (ne od nule)!

22

22

## Povezivanje parametara - primjer 1

```
$upit=$db->prepare ("INSERT INTO kategorija "
    . " (kategorijaID, nazivKat) VALUES (?, ?)");
$kategorija=array(3,'čokolada');
$upit->bindValue(1,$kategorija[0]);
$upit->bindValue(2,$kategorija[1]);
$upit->execute();

$kategorija=array(4,'keksi');
$upit->bindValue(1,$kategorija[0]);
$upit->bindValue(2,$kategorija[1]);
$upit->execute();
```

23

23

## Povezivanje parametara - primjer 2

```
$upit=$db->prepare("INSERT INTO kategorija (kategorijaID, nazivKat) "
    . "VALUES (:kategorijaID, :nazivKat)");
$upit->bindValue(":kategorijaID",3);
$upit->bindValue(":nazivKat",'čokolada');
$upit->execute();

$upit->bindValue(":kategorijaID",4);
$upit->bindValue(":nazivKat",'keksi');
$upit->execute();
```

24

24

## Povezivanje parametara - bindValue

```
$upit=$db->prepare ("INSERT INTO kategorija "
    . "(kategorijaID, nazivKat) VALUES (?, ?)");
$upit->bindParam(1,$katID);
$upit->bindParam(2,$naziv);

$katID=4;
$naziv='keksi';
$upit->execute();
```

25

25

## Promjena strukture tablice Proizvod

```
$upit=$db->prepare("ALTER TABLE proizvod ADD dobavljacID INT NOT
NULL");
$upit->execute();

$upit=$db->prepare("ALTER TABLE proizvod ADD kategorijaID INT
NOT NULL");
$upit->execute();
```

26

26

## Unos podataka u tablicu proizvod

```
$upit= $db->prepare("INSERT INTO proizvod "
    . "(proizvodID, nazivPro, cijena,kolicina, "
    . "dobavljacID, kategorijaID) "
    . "VALUES (?, ?, ?, ?, ?, ?)");

$proizvod=array(1, "Oliver Futura", 34.99,25,2,5);
$upit->execute($proizvod);

$proizvod=array (2, "Vegeta pikant", 12.50,100,3,2);
$upit->execute($proizvod);

$proizvod=array(3, "Dorina Mousse", 7.05,70,1,3);
$upit->execute($proizvod);

$proizvod=array(4, "Životinjsko carstvo", 1.45,150,1,3);
$upit->execute($proizvod);
```

27

27

## Naredba SELECT

- ▶ dohvaćanje podataka iz baze obavlja se naredbom SELECT

```
SELECT imena stupaca
FROM imena tablica
WHERE uvjeti
ORDER BY stupci
LIMIT pomak, broj
```
- ▶ u SELECT dijelu može se koristiti \* umjesto da se navode pojedinačno svi stupci
- ▶ u FROM dijelu se navode imena tablica odnosno logička veza (JOIN) potrebna za dohvaćanje podataka
- ▶ u WHERE dijelu navodi se uvjet (ili više njih)
- ▶ u ORDER BY navode se stupci sortiranja (uzlazno: ASC, silazno: DESC)
- ▶ u LIMIT dijelu navodi se pomak i broj redova koji će naredba vratiti
  - npr. LIMIT 8, 10 će dati redove od 9 do 18 ali
  - LIMIT 6 daje prvih 6 redova!

28

28

## Dohvaćanje podatka iz baze

- ▶ elegantna šetnja naredbom foreach

```
$upit->execute();  
foreach($upit as $red) { ... }
```

- ▶ tip varijable \$red ovisi o načinu dohvaćanja

- PDO::FETCH\_NAMED
- PDO::FETCH\_BOUND
- PDO::FETCH\_BOTH
- PDO::FETCH\_OBJ

29

29

## Dohvaćanje podataka iz baze

```
$db= new PDO  
('mysql:host=localhost;dbname=trgovina','root','');  
$db->exec("SET NAMES utf8");  
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);  
  
$upit=$db->prepare('SELECT nazivPro,nazivKat,nazivDob FROM  
(proizvod p inner join kategorija k on  
p.kategorijaID=k.kategorijaID)  
INNER JOIN Dobavljac D on p.dobavljacID=D.dobavljacID');  
$upit->setFetchMode(PDO::FETCH_NAMED);  
$upit->execute();  
  
foreach ($upit as $red){  
    echo $red["nazivPro"] . " | ". $red["nazivKat"] . " | " .  
$red["nazivDob"] . "<br/>";  
}
```

Vepta pikant | dodatak jelu | Podravka  
Oliver Futura | deterdžent | Labud  
Dorina Mousse | čokolada | Kraš  
Životinjsko carstvo | čokolada | Kraš

30

15

## Dohvaćanje podataka iz baze

```
$db= new PDO  
('mysql:host=localhost;dbname=trgovina','root','');  
$db->exec("SET NAMES utf8");  
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);  
  
$upit=$db->prepare('SELECT nazivPro,nazivKat,nazivDob FROM  
(proizvod p inner join kategorija k on  
p.kategorijaID=k.kategorijaID)  
INNER JOIN Dobavljac D on p.dobavljacID=D.dobavljacID');  
$upit->setFetchMode(PDO::FETCH_BOUND);  
$upit->bindColumn("nazivPro",$nazivPro);  
$upit->bindColumn("nazivKat",$nazivKat);  
$upit->bindColumn("nazivDob",$nazivKat);  
$upit->execute();  
  
foreach ($upit as $red){  
    echo $nazivPro . " | ". $nazivKat . " | " . $nazivKat .  
"<br/>";  
}
```

31

31

## Dohvaćanje podataka iz baze

```
$db= new PDO  
('mysql:host=localhost;dbname=trgovina','root','');  
$db->exec("SET NAMES utf8");  
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);  
  
$upit=$db->prepare('SELECT nazivPro,nazivKat,nazivDob FROM  
(proizvod p inner join kategorija k on  
p.kategorijaID=k.kategorijaID)  
INNER JOIN Dobavljac D on p.dobavljacID=D.dobavljacID');  
$upit->setFetchMode(PDO::FETCH_OBJ);  
$upit->execute();  
  
foreach ($upit as $red){  
    echo $red->nazivPro . " | ". $red->nazivKat . " | " .  
        $red->nazivDob . "<br/>";  
}
```

32

32

## Metode fetch\*

- ▶ `fetch()` – vraća sljedeći red u rezultatu
- ▶ `fetchAll()` – vraća niz sa svim rezultatima
- ▶ `fetchColumn($stupac)` – vraća vrijednost u željenom stupcu idućeg retka
  - `$nazivDob=$upit(fetchColumn('nazivDob'));`

33

33

## Metoda `fetch()`

```
$db= new PDO  
('mysql:host=localhost;dbname=trgovina','root','');  
$db->exec("SET NAMES utf8");  
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);  
  
$upit=$db->prepare('SELECT nazivPro,nazivKat,nazivDob FROM  
(proizvod p inner join kategorija k on  
p.kategorijaID=k.kategorijaID)  
INNER JOIN Dobavljac D on p.dobavljacID=D.dobavljacID');  
$upit->setFetchMode(PDO::FETCH_OBJ);  
$upit->execute();  
  
while ($red=$upit->fetch()) {  
    echo $red->nazivPro. " | ". $red->nazivKat . " | " .  
        $red->nazivDob . "<br/>";  
}
```

34

34

## Metoda fetchAll()

```
$db= new PDO  
('mysql:host=localhost;dbname=trgovina','root','');  
$db->exec("SET NAMES utf8");  
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);  
  
$upit=$db->prepare('SELECT nazivPro,nazivKat,nazivDob FROM  
(proizvod p inner join kategorija k on  
p.kategorijaID=k.kategorijaID)  
INNER JOIN Dobavljac D on p.dobavljacID=D.dobavljacID');  
$upit->setFetchMode(PDO::FETCH_OBJ);  
$upit->execute();  
var_dump($upit->fetchAll());  
  
array(4) { [0]=> object(stdClass)#3 (3) { ["nazivPro"]=> string(13) "Vegeta pikant"  
["nazivKat"]=> string(12) "dodatak jelu" ["nazivDob"]=> string(8) "Podravka" } [1]=>  
object(stdClass)#4 (3) { ["nazivPro"]=> string(13) "Oliver Futura" ["nazivKat"]=>  
string(11) "deterđent" ["nazivDob"]=> string(5) "Labud" } [2]=> object(stdClass)#5 (3) {  
["nazivPro"]=> string(13) "Dorina Mousse" ["nazivKat"]=> string(9) "čokolada"  
["nazivDob"]=> string(5) "Kraš" } [3]=> object(stdClass)#6 (3) { ["nazivPro"]=> string(20)  
"Životinjsko carstvo" ["nazivKat"]=> string(9) "čokolada" ["nazivDob"]=> string(5) "Kraš"  
}  
}
```

35

35

## Dohvaćanje podataka iz baze

```
$db= new PDO  
('mysql:host=localhost;dbname=trgovina','root','');  
$db->exec("SET NAMES utf8");  
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);  
$upit=$db->prepare('SELECT nazivPro,nazivKat,nazivDob FROM  
(proizvod p inner join kategorija k on  
p.kategorijaID=k.kategorijaID)  
INNER JOIN Dobavljac D on p.dobavljacID=D.dobavljacID');  
$upit->setFetchMode(PDO::FETCH_OBJ);  
$upit->execute();  
echo '<table border="1">';  
foreach ($upit as $red){  
    echo '<tr>';  
    echo '<td>' . $red->nazivPro . '</td>';  
    echo '<td>' . $red->nazivKat . '</td>';  
    echo '<td>' . $red->nazivDob . '</td>';  
    echo "<tr/>";  
}  
echo '</table>';
```

Dorina Mousse	čokolada	Kraš
Zivotinjsko carstvo	čokolada	Kraš
Oliver Futura	deterđent	Labud
Vegeta pikant	dodatak jelu	Podravka

36

36

## Kombiniranje PHP-a i HTML-a

```
<?php
$db= new PDO ('mysql:host=localhost;dbname=trgovina','root','');
$db->exec("SET NAMES utf8");
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$upit=$db->prepare('SELECT
nazivPro,kolicina,cijena,kolicina*cijena as vrijednost FROM
proizvod');
$upit->setFetchMode(PDO::FETCH_OBJ);
$upit->execute();
?>
<div style="text-align: center;">
<h3>Baza proizvoda</h3>
<table border="1" cellpadding="2" cellspacing="2"
style="width:60%; margin-left: auto; margin-right: auto;">
<tr>
<th>Naziv proizvoda</th>
<th>Količina</th>
<th>Cijena</th>
<th>Vrijednost robe</th>
</tr>
```

37

## Kombiniranje PHP-a i HTML-a (nastavak)

```
<?php
$brojpro=0;
foreach ($upit as $red) {
    echo '<tr>';
    echo '<td>' . $red->nazivPro . '</td>';
    echo '<td>' . $red->kolicina . '</td>';
    echo '<td>' . $red->cijena . '</td>';
    echo '<td>' . $red->vrijednost . '</td>';
    echo "<tr/>";
    $brojpro++;
}
?>
</table>
<p><?php echo "Broj proizvoda:" . $brojpro; ?> </p>
</div>
```

38

## ...nastavak

Baza proizvoda

Naziv proizvoda	Količina	Cijena	Vrijednost robe
Oliver Futura	25	34.99	874.75
Vegeta pikant	100	12.50	1250.00
Dorina Mousse	70	7.05	493.50
Životinjsko carstvo	150	1.45	217.50

Broj proizvoda: 4

39

39

## Korištenje HEREDOC-a

```
$db= new PDO ('mysql:host=localhost;dbname=trgovina','root','','');
$db->exec("SET NAMES utf8");
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$upit=$db->prepare('SELECT
nazivPro,kolicina,cijena,kolicina*cijena as vrijednost FROM
proizvod');
$upit->setFetchMode(PDO::FETCH_OBJ);
$upit->execute();
$tablica=<<<HTML
<div style="text-align: center;">
<h3>Baza proizvoda</h3>
<table border="1" cellpadding="2" cellspacing="2"
style="width:60%; margin-left: auto; margin-right: auto;">
<tr>
<th>Naziv proizvoda</th>
<th>Količina</th>
<th>Cijena</th>
<th>Vrijednost robe</th>
</tr>
```

40

## Korištenje HEREDOC-a

```
$brojpro=0;
foreach ($upit as $red) {
    $stablica .= <<<HTM
    echo '<tr>';
    echo '<td>' . $red->nazivPro . '</td>';
    echo '<td>' . $red->kolicina . '</td>';
    echo '<td>' . $red->cijena . '</td>';
    echo '<td>' . $red->vrijednost . '</td>';
    echo "<tr/>";
    $brojpro++;
HTM;
}
$stablica.= <<<HTM
</table>
<p> Broj proizvoda: $brojpro </p>
</div>
HTM;
echo $stablica;
?>
```

41

## Ažuriranje (UPDATE) i brisanje (DELETE)

```
<?php
$db= new
PDO('mysql:host=localhost;dbname=trgovina','root','');
$db->exec("SET NAMES utf8");
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

$upit=$db->prepare('UPDATE Proizvod SET kolicina=kolicina+10
WHERE proizvodID=1');
$upit->execute();

$upit=$db->prepare('DELETE FROM Kategorija
WHERE nazivKat="sok"');
$upit->execute();
?>
```

42

42

## SQL injection

```
$korisnik = $_POST['user'];
$sifra= $_POST['passw'];
$upit = "SELECT * FROM korisnici WHERE user='$korisnik'
        AND password='$sifra'";
```

- ▶ ako korisnik uneše *ivoivic* i *mojasifra* sve je ok

```
SELECT * FROM korisnici WHERE user='ivoivic'
        AND password='mojasifra'
```

- ▶ ako korisnik uneše *admin'#* za ime korisnika i ništa za šifru?

```
SELECT * FROM korisnici WHERE user='admin' #' AND     password=''
```

43

43

## SQL injection

```
$korisnik = $_POST['user'];
$sifra= $_POST['passw'];
$upit = "DELETE FROM korisnici WHERE user='$korisnik'
        AND password='$sifra'";
```

- ▶ ako korisnik uneše *bilosto' OR 1=1 #* za ime korisnika?

```
DELETE FROM korisnici
WHERE user='bilosto' OR 1=1 #' AND     password=''
```

44

44



## P9: Rad s datotekama

Kolegij: Programiranje u PHP  
Predavač: Dalibor Bužić

### Rad s datotekama

- ▶ PHP omogućuje rad s datotekama i direktorijima pohranjenim na web poslužitelju, kao jedan od načina čuvanja informacija izvan samih skripti
- ▶ direktorij je u osnovi također datoteka, a služi pohrani drugih datoteka
- ▶ direktoriji imaju hijerarhijsko uređenje → osnovni (ishodišni) je root (top-level) direktorij u kojem se stvaraju drugi direktoriji unutar njih neki treći itd.
- ▶ datoteke mogu pohranjivati raznovrsne podatke, ali i neke podatke o samoj datoteci (npr. tko je vlasnik, kad je datoteka stvorena)
- ▶ PHP posjeduje funkcije koje omogućuju dohvaćanje informacija o datotekama, te otvaranje, čitanje i zapisivanje podataka

## Rad s datotekama

- ▶ putanje do datoteke se različito označavaju na Windows u odnosu na UNIX platformi
- ▶ elementi putanje se razdvajaju kosom crtom (/) na UNIX sustavima, a obrnutom kosom crtom na Windows sustavima:
  - /home/php/predavanja/p1.txt (UNIX)
  - c:\home\php\predavanja\p1.txt (Windows)
    - Ovisno o konfiguraciji, moguće je da trebaju dvije obrnute kose crte zaredom c:\\xampp\\htdocs\\p1.txt
- ▶ PHP na Windowsima u pravilu automatski pretvara / u \

3

3

## Dohvaćanje informacija o datoteci

- ▶ file\_exists() – ispituje postoji li datoteka (npr. prije otvaranja datoteke dobro je napraviti ovakvu provjeru
  - funkcija vraća vrijednost true ako postoji datoteka u navedenoj putanji, inače vraća false
  - `file_exists("/home/php/labosi/lab1.txt")`
- ▶ filesize() – vraća veličinu datoteke u bajtovima, ili false u slučaju greške
  - `filesize("/home/php/labosi/lab1.txt")`
- ▶ basename() – iz cijele putanje vraća ime datoteke s ili bez sufiksa ili ime najdubljeg direktorija
  - `$d= basename("/home/php/labosi/lab1.txt") //lab1.txt`
  - `$d= basename("/home/php/labosi") //labosi`
  - `$d= basename("/home/php/labosi/lab1.txt",".txt") //lab1`

4

4

## Dohvaćanje informacija o datoteci

- ▶ PHP ima tri funkcije za datoteke koje su povezane s vremenom:
  - fileatime() – vraća vrijeme kad je zadnji put pristupljeno datoteci tj. kad joj je zadnji put čitan sadržaj
  - filectime() – vraća vrijeme kad su zadnji put mijenjani metapodaci npr. kad je datoteka stvorena ili kad su promijenjene dozvole pristupa
  - filemtime() – vraća vrijeme kad je zadnji put mijenjan sadržaj datoteke
- ▶ vrijeme je izraženo u UNIX timestamp (vremenski žig) formatu: cjelobrojna vrijednost broj sekundi koji je protekao između 1.1.1970. i vremena promjene datoteke

5

5

## Otvaranje datoteke

- ▶ u radu s datotekama uobičajeno je najprije otvoriti datoteku funkcijom fopen() cime se stvara hvataljka (file handler, pointer) pridružena toj datoteci
  - hvataljka je podatkovnog tipa resource
  - neke funkcije PHP-a ne zahtijevaju otvaranje datoteke

```
$hvata=fopen("./lab1.txt", "r")
```

- ▶ prvi argument može biti:
  - samo ime datoteke (npr "lab1.txt") za datoteku u tekućem direktoriju
  - relativna putanja datoteke (npr "./lab1.txt") ili
  - apsolutna putanja datoteke (npr "/dokumenti/lab1.txt")
- ▶ moguće je specificirati datoteku na udaljenom serveru (samo za čitanje):

```
$hvata=fopen("http://www.vsite.hr/?q=hr/node/514", "r")
```

6

6

## Otvaranje datoteke

- ▶ drugi argument nalaže funkciji fopen() kako će se datoteka koristiti; moguće vrijednosti su:

vrijednost	opis
r	otvara datoteku samo za čitanje
r+	otvara datoteku za čitanje i pisanje
w	otvara datoteku samo za pisanje; ako datoteka ima neki sadržaj, on će biti izgubljen; ako datoteka ne postoji PHP će ju pokušati kreirati
w+	otvara datoteku za čitanje i pisanje; ako datoteka ima neki sadržaj, on će biti izgubljen; ako datoteka ne postoji PHP će ju pokušati kreirati
a	otvara datoteku samo za dodavanje; podaci se dodaju na kraj datoteke; ako datoteka ne postoji PHP će ju pokušati kreirati
a+	otvara datoteku za čitanje i dodavanje; podaci se dodaju na kraj datoteke; ako datoteka ne postoji PHP će ju pokušati kreirati

7

7

## Otvaranje i zatvaranje datoteke

- ▶ ako je došlo do problema prilikom otvaranja datoteke, fopen() umjesto hvataljke vraća false
- ▶ operacije s datotekama i direktorijima su podložne greškama pa je dobro na neki način provjeriti je li došlo do greške (npr. zbog toga što datoteka ne postoji, ili skripta nema prava pristupa datoteci i sl.)

```
if (!($hvati=fopen("./lab1.txt", "r"))) die("Nije  
moguće otvoriti datoteku");
```

- ▶ umjesto izlaska pomoći die(), moguće je podići grešku ili hvatati iznimku
- ▶ zatvaranje datoteke obavlja se pozivom funkcije fclose()

```
fclose($hvati);
```

8

8

## Čitanje i pisanje nizova znakova

- ▶ funkcija fread() čita niz znakova iz datoteke

```
$hvat=fopen("podaci.txt", "r"));  
$pod=fread($hvat, 10);
```
- ▶ prvi argument je hvataljka datoteke koju čitamo, a drugi argument je broj znakova koje želimo pročitati
- ▶ funkcija vraća niz znakova (string)
- ▶ nakon čitanja pokazivač u datoteci se pomiče za broj pročitanih znakova (u primjeru gore to je 10)
- ▶ ako se ponovno pozove funkcija fread(), pročitat će sljedećih 10 znakova
- ▶ ako ima manje od 10 znakova, pročitat će koliko ih već ima
- ▶ čitanje jednog znaka može se obaviti pozivom funkcije fgetc()

```
$pod=fgetc($hvat);
```
- ▶ fgetc() je spor u radu s velikim datotekama

9

## Čitanje i pisanje nizova znakova

- ▶ upis podataka u datoteku može se obaviti pomoću funkcije fwrite()
- ▶ funkcija prihvata dva argumenta: hvataljku i niz znakova koji treba upisati
- ▶ funkcija fwrite() vraća broj znakova koji su upisani u datoteku (ili false ako upis ne uspije)

```
$hvat=fopen("podaci.txt", "w");  
fwrite($hvat, "Snjeguljica");  
fwrite($hvat, " i sedam patuljaka");
```
- ▶ funkcija može primiti i treći argument, a on predstavlja maksimalni broj upisanih znakova  

```
fwrite($hvat, "Snjeguljica", 5);
```
- ▶ zato gornja naredba upisuje Snjeg u datoteku
- ▶ ako se niz znakova želi zapisati kao jedan redak, potrebno je na kraj dodati \n  

```
fwrite($hvat, "Snjeguljica\n");
```

10

10

# Čitanje i pisanje u datoteku

- ▶ čitanje jednog reda datoteke može se obaviti pomoću fgets() funkcije
- ▶ funkcija vraća string koji uključuje znak(ove) za kraj reda
- ▶ za čitanje CSV datoteka služi funkcija fgetcsv()
- ▶ argument funkcije je hvataljka, a opcionalni argumenti su:
  - najveći broj znakova za čitanje
  - delimiter (podrazumijevani je zarez; ako se koristi tab-separated-values datoteka, potrebno je specificirati "\t")
  - znak za ograničavanje string vrijednosti (podrazumijevano je dvostruki navodnik)
  - znak za escape sekvence (podrazumijevano je obrnuta kosa crta \)
- ▶ fgetcsv() vraća indeksirani niz

11

11

# Čitanje i pisanje u datoteku

```
$hva=fopen("./drzave.csv","r");  
while($zapis=fgetcsv($hva,100,";"))  
{  
    echo "Kratica: {$zapis[0]} Naziv: {$zapis[1]} <br />" ;  
}  
fclose($hva);
```



drzave.csv - Notepad	
File	Edit
ALB	Albanija
AUT	Austrija
BEL	Belgija
BGR	Bugarska
BIH	Bosna i Hercegovina
BLR	Bjelorusija
CUB	Kuba
CZE	Češka
CYP	Cipar
DNK	Danska
ESP	Španjolska
EST	Estonija
FIN	Finska
FRA	Francuska
GBR	Velika Britanija
GRC	Grčka
HUN	Hrvatska
IRL	Irska
ITA	Italija
LUX	Luksemburg
LTU	Litva
LVA	Latvija
MKD	Makedonija
MLT	Malta
MNG	Mongolija
NLD	Nizozemska
POL	Poљska
PRK	DNR Koreja
PRT	Portugal
ROM	Rumunjska
RUS	Rusija
SRB	Srbija
MNE	Crongora
SVK	Slovačka
SVN	Slovenija
SWE	Svedska

12

12

## Odmak od slijednog čitanja datoteke

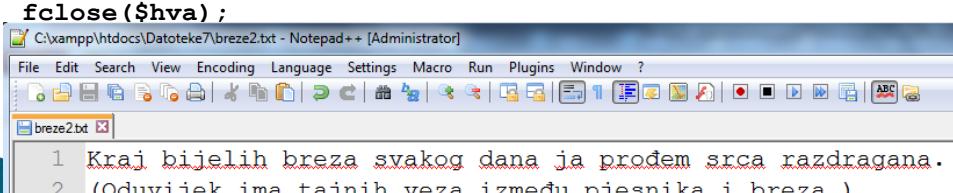
- ▶ dosad izložene funkcije manipuliraju podacima slijedno
- ▶ 'skakanje' po sadržaju datoteke moguće je funkcijama
  - fseek() – postavlja pokazivač datoteke na određenu točku u datoteci
  - rewind() – postavlja pokazivač na početak datoteke
  - ftell() – vraća trenutnu poziciju datotečnog pokazivača

13

13

## Odmak od slijednog čitanja datoteke

```
if( !($hva=fopen("breze2.txt","r")) )  
    die("Ne mogu otvoriti datoteku!");  
echo ftell($hva) . "<br>"; //ispisuje:    
fseek($hva, 10);  
echo fread($hva, 5) . "<br>"; //ispisuje:    
echo ftell($hva) . "<br>"; // ispisuje:    
fseek($hva, ftell($hva) - 9);  
echo fread($hva, 3) . "<br>"; ispisuje:    
fclose($hva);
```



The screenshot shows a Notepad++ window titled 'breze2.txt'. The code is displayed in the editor area. Below the editor, the status bar shows the path 'C:\xampp\htdocs\Datoteke\breze2.txt - Notepad++ [Administrator]'. The output pane at the bottom displays the results of the script's execution:

```
1 Kraj bijelih breza svakog dana ja prodem srca razdragana.  
2 (Oduvijek ima tajnih veza između pjesnika i breza.)  
3
```

14

14

## Čitanje i pisanje cijelih datoteka

- ▶ pristup cijelom sadržaju datoteke odjednom moguć je pomoću funkcija:
  - `file()` – učitava cijelu datoteku u niz, bez naredbe otvaranja
    - svaki element niza je jedan red datoteke
    - može otvoriti datoteku s udaljenog poslužitelja
  - `file_get_contents()` i `file_put_contents()` – čita i upisuje sadržaj datoteke bez naredbe otvaranja
    - `file_get_contents()` vraća kompletan sadržaj u jednu varijablu
  - `fpassthru()` – prikazuje sadržaj otvorene datoteke
    - prikazuje sadržaj datoteke direktno u web preglednik
  - `readfile()` – prikazuje sadržaj bez naredbe otvaranja
    - prikazuje sadržaj datoteke direktno u web preglednik
- ▶ gore navedene funkcije treba koristiti za relativno male datoteke (do najviše par MB)
- ▶ ako radimo s datotekom od npr. 100MB uputnije je koristiti funkcije `fread()` i `fgets()`

15

15

## Čitanje i pisanje cijelih datoteka

```
readfile("http://prognoza.hr/sedam.php?id=sedam&param=Hrvatska&code=14240");
```

Dan	Ponедјелjak 13.01.2014.	Уторак 14.01.2014.	Сrijeda 15.01.2014.	Четвртак 16.01.2014.	Петак 17.01.2014.	Субота 18.01.2014.	Недјеља 19.01.2014.
	Tmin 2 °C 36°F	Tmax 10 °C 50°F					
Ponedjeljak 13.01.2014.	2 °C 36°F	10 °C 50°F					
Уторак 14.01.2014.	4 °C 39°F	10 °C 50°F					
Сrijeda 15.01.2014.	3 °C 37°F	7 °C 45°F					
Четвртак 16.01.2014.	0 °C 32°F	7 °C 45°F					
Петак 17.01.2014.	0 °C 32°F	11 °C 52°F					
Субота 18.01.2014.	3 °C 37°F	14 °C 57°F					
Недјеља 19.01.2014.	5 °C 41°F	12 °C 54°F					

16

16

## Uključivanje PHP koda jedne datoteke u drugu

- ▶ uključivanje PHP koda jedne datoteke u drugu moguće je naredbama include() i require()
- ▶ uključivanje koda jedne datoteke u drugu daje isti rezultat kao da je pozvana skripta kopirana u pozivajuću skriptu na mjestu poziva
- ▶ naredbe se mogu pisati s ili bez zagrada

```
include("header.php");  
include "header.php";
```
- ▶ razlika između include() i require() je sljedeća: ako se specificirana datoteka ne može uključiti, include() će generirati upozorenje i nastaviti s izvođenjem, dok će require() generirati fatalnu grešku i prekinuti izvođenje skripte
- ▶ include\_once i require\_once uključuju datoteku samo jednom, čak i ako se pozove više puta, čime se izbjegavaju konflikti

17

17

## Primjer: zaglavje.php (...)

```
<html>  
    <head>  
        <meta charset="UTF-8">  
        <title><?php echo $stranica['naslov'];?></title>  
    </head>  
    <body>  
        <table width="90%" border="0" cellspacing="4"  
            cellpadding="4">  
            <tr> <td><a href="#">Home</a></td>  
                <td><a href="#">Sitemap</a>  
                </td> <td><a href="#">Search</a></td>  
                <td><a href="#">Help</a></td>  
            </tr>  
        </table>  
    </body>  
</html>
```

18

18

## Primjer: (...) podnozje.php (...)

```
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
  </head>
  <body>
    <br>
    <p style="text-align:center"> Copyright 2013 -
      <?php date_default_timezone_set('Europe/Zagreb') ;
      echo date("Y", mktime()); ?> </p>
  </body>
</html>
```

19

19

## Primjer: (...) glavna.php

```
<?php

$stranica = array();
$stranica['naslov'] = 'Katalog proizvoda';

// uključivanje zaglavlja
include('zaglavje.php');
?>

<!-- Sadržaj stranice --&gt;

&lt;?php // uključivanje podnožja
include('podnozje.php');
?&gt;</pre>
```

20

20

## Rad s dozvolama

- ▶ funkcija chmod() omogućuje promjenu dozvola korištenja datoteke ili direktorija na UNIX baziranim web poslužiteljima  
`chmod ("podaci.txt", 0644);`
- ▶ vodeća nula je bitna jer govori PHP-u da se radi o oktalnoj vrijednosti
- ▶ prva znamenka određuje dozvole vlasnika, druga dozvole grupe, a treća dozvole svima

vrijednost	opis
0	nikakve dozvole
1	može samo izvršiti datoteku
2	može samo upisivati u datoteku
3	može pisati i izvršiti datoteku
4	može samo čitati datoteku
5	može čitati i izvršiti datoteku
6	može čitati i pisati u datoteku
7	može čitati, pisati i izvršiti datoteku

21

21

## Kopiranje, preimenovanje i brisanje datoteke

- ▶ kopiranje se obavlja pomoću funkcije copy(izvor, odrediste)  
`copy ("podaci.txt", "kopija.txt");`
- ▶ preimenovanje se obavlja pomoću funkcije rename(staro\_ime, novo\_ime)  
`rename ("podaci.txt", "podaci.bkp");`
- ▶ brisanje se obavlja pomoću funkcije unlink(datoteka)  
`unlink ("podaci.bkp");`
- ▶ prije poziva ovih funkcija dobro je provjeriti postoji li datoteka pomoću file\_exists() funkcije

22

22

## Rad s direktorijima

- ▶ opendir() funkcijom otvara se direktorij i stvara se hvataljka  

```
$hvat=opendir("/home/vsite");
```
- ▶ zatvaranje direktorija:  

```
closedir($hvat);
```
- ▶ dohvaćanje sljedećeg imena datoteke obavlja se pomoću readdir() funkcije  

```
$imedat=readdir($hvat);
```
- ▶ chdir() mijenja trenutni direktorij u novi
- ▶ mkdir() stvara novi direktorij
- ▶ rmdir() briše postojeći direktorij (mora biti prazan, i jasno potrebno je imati adekvatnu dozvolu)
- ▶ is\_dir() vraća true ako je argument direktorij
- ▶ is\_file() vraća true ako je argument obična datoteka

23



## P10: Rad s vremenskim podacima

Kolegij: Programiranje u PHP  
Predavač: Dalibor Bužić

### Problemi vremenskih podataka u realnom životu

- ▶ Koji datum i vrijeme je bio 256 sati, 12 minuta i 40 sekundi prije 28. prosinca 2013. godine?
- ▶ mjeseci imaju različit broj dana
- ▶ tjedni se ne poklapaju s počecima ni mjeseca ni godine
- ▶ koji je prvi dan u tjednu?
- ▶ vremenske zone, ljetno i zimsko računanje vremena, južna i sjeverna hemisfera, datumska granica
- ▶ način prikaza i lokalizacija:
  - 9:45:30 p.m., 19 December 2013
  - 2013-12-19 21:45:30
  - 19 December, 2013, 9:45:30 p.m.
  - 19.12.2013. 21:45:30
  - četrnaest i pol minuta prije deset navečer u četvrtak

## Prikaz datuma i vremena u PHP-u

```
echo time(); // 1390267848
```

- ▶ vrijeme je izraženo u broju sekundi proteklih od 1.1.1970. 00:00:00 (*Unix epoha*)
- ▶ dobiveno je vrijeme ono sa servera, a ne klijenta
- ▶ budući da je za većinu korisnika ovo vrijeme razumljivo kao i klinasto pismo, potrebno ga je prikazati u ljudskom oku ugodnom obliku
- ▶ za to se koristi funkcija date(format[, timestamp]) koja u zadanim formatu pokazuje trenutno vrijeme ili vrijeme koje je navedeno kao opcionalni argument

3

3

## Formatiranje prikaza datuma i vremena

Znak	Opis
<b>DAN</b>	
d	dan u mjesecu s vodećom nulom
j	dan u mjesecu bez vodeće nule
S	sufiks dana (eng – st,nd,th), uglavnom se koristi s j
I (malo slovo L)	naziv dana u tjednu
D	kratka za dan u tjednu (Mon, Tue...)
w	redni broj dana u tjednu (0=nedjelja!)
<b>MJESEC</b>	
F	puno ime mjeseca (January, February...)
M	trošlovna kratka za ime mjeseca
m	brojčani prikaz mjeseca s vodećom nulom
n	brojčani prikaz mjeseca bez vodeće nule
<b>GODINA</b>	
y	godina – dvije znamenke
Y	godina – četiri znamenke

4

4

## Formatiranje prikaza datuma i vremena

Znak	Opis
<b>SAT</b>	
h	sat u 12-satnom formatu s vodećom nulom
g	sat u 12-satnom formatu bez vodeće nule
H	sat u 24-satnom formatu s vodećom nulom
G	sat u 24-satnom formatu bez vodeće nule
a	am/pm malim slovima
A	AM/PM velikim slovima
O (veliko slovo o)	razlika lokalnog vremena prema GMT/UTC
<b>MINUTA</b>	
i	minute s vodećom nulom
j	minute bez vodeće nule
<b>SEKUNDA</b>	
s	sekunde s vodećom nulom
Z	sekunde bez vodeće nule

5

5

## Formatiranje prikaza datuma i vremena

Znak	Opis
<b>POTPUNI DATUM I VRIJEME</b>	
c	ISO-8601 format (YYYY-MM-DDTHH:MM:SS±HH:MM)
r	RFC-2822 format (WWW, DD MMM YYYY HH:MM:SS±HHMM)
U	broj sekundi od početka UNIX epohe

```
$datvri=time();
$oblici=array('U', 'r','c','l, F,jS, g:i A',
'H:i:s D d M y', 'm/j/y g:i:s O (T)');
foreach($oblici as $format){
    echo "<b>Obljek:</b> " . date($format,$datvri) . "<br />
<br />";
}
```

```
U: 1390077636
r: Sat, 18 Jan 2014 20:40:36 +0000
c: 2014-01-18T20:40:36+00:00
l, F,jS, g:i A: Saturday, January, 18th, 8:40 PM
H:i:s D d M y: 20:40:36 Sat 18 Jan 14
m/j/y g:i:s O (T): 01/18/14 8:40:36 +0000 (UTC)
```

6

6

## Prikaz dana i mjeseca na hrvatskom jeziku

- ▶ funkcija setlocale() prima dva argumenta: kategoriju i kod jezika
- ▶ kategorija je konstanta, što se tiče datuma može se upotrijebiti LC\_ALL ili LC\_TIME
- ▶ funkciju iconv() pozovemo ako ne želimo gledati maštovite zamjene za naše dijakritike

```
setlocale(LC_ALL,'croatian');
$vrzig= mktime(18,45,15,1,21,2014);
echo date('r (T)', $vrzig) . '<br />';
echo strftime('%A, %d. %B %Y %H:%M:%S',$vrzig). '<br />';
echo iconv('ISO-8859-2', 'UTF-8',strftime('%A, %d. %B %Y
%H:%M:%S',$vrzig)) ;
```

```
Tue, 21 Jan 2014 18:45:15 +0000 (UTC)
utorak, 21. siječanj 2014 18:45:15
utorak, 21. siječanj 2014 18:45:15
```

7

## strftime() i time()

```
print'funkcija strftime():  ';
print strftime('Danas je %d.%m.%y a trenutno vrijeme je
%I:%M:%S');
print "\n";
print'funkcija date():  ';
print 'Danas je ' . date('d.m.y') . ' a trenutno vrijeme
je ' .date('h:i:s');
```

```
funkcija strftime(): Danas je 07.01.15 a trenutno vrijeme je 02:59:50
funkcija date(): Danas je 07.01.15 a trenutno vrijeme je 02:59:50
```

8

8

## Stvaranje datuma i vremena pomoću `mktime()`

- ▶ funkcija `mktime()` prima argumente i iz njih stvara datum i vrijeme u UNIX timestamp formatu
- ▶ argumenti su (poredani po ugradnji):
  - sati (0–23)
  - minute (0–59)
  - sekunde (0–59)
  - mjesec (1–12)
  - dan (1–28|29|30|31)
  - godina (4 znamenke)
  - ljetno\_vrijeme (true ili 1 ako je ljetno, false ili 0 ako je zimsko)

```
$vrzig= mktime(18,45,15,1,21,2013);  
echo date('r (T)', $vrzig);
```

```
Mon, 21 Jan 2013 18:45:15 +0000 (UTC)
```

9

9

## Pretvaranje govornih izraza u datume

- ▶ funkcija `strtotime()` prima tekstualni argument i iz njega stvara datum i vrijeme u UNIX timestamp formatu
- ▶ prihvatljivi argumenti
  - dvije/četiri znamenke za godinu
  - jedna/dvije znamenke za dan odnosno mjesec
  - imena mjeseca (potpuno ime, tro ili četveroslovna kratica)
  - naziv dana u tjednu (potpuni ili troslovna kratica)
  - numerički datumi u formatu [[yy]y]y-[m]m-[d]d ili [m]m/[d]d/[yy]yy
  - dopuštene su jedinice kao hour, day, week, month, year
  - dopušteno je i now, last, today, yesterday, ago i još neke

10

10

## Pretvaranje govornih izraza u datume

```
<table>
<?php
$mojavremena=array("now", "today",
"tomorrow", "yesterday", "Wednesday",
"this Sunday", "last Sunday",
"+20 minutes", "+5 hours",
"-2 months", "20 seconds",
"next month", "last week",
"last year", "5 weeks ago");
foreach($mojavremena as $mojav) {
    echo "<tr><td>$mojav</td><td>" .
        date('r',strtotime($mojav)) . "</td></tr>";
}
?>
</table>
```

now	Tue, 21 Jan 2014 17:58:05 +0000
today	Tue, 21 Jan 2014 00:00:00 +0000
tomorrow	Wed, 22 Jan 2014 00:00:00 +0000
yesterday	Mon, 20 Jan 2014 00:00:00 +0000
Wednesday	Wed, 22 Jan 2014 00:00:00 +0000
this Sunday	Sun, 26 Jan 2014 00:00:00 +0000
last Sunday	Sun, 19 Jan 2014 00:00:00 +0000
+20 minutes	Tue, 21 Jan 2014 18:18:05 +0000
+5 hours	Tue, 21 Jan 2014 22:58:05 +0000
-2 months	Thu, 21 Nov 2013 17:58:05 +0000
20 seconds	Tue, 21 Jan 2014 17:58:25 +0000
next month	Fri, 21 Feb 2014 17:58:05 +0000
last week	Mon, 13 Jan 2014 17:58:05 +0000
last year	Mon, 21 Jan 2013 17:58:05 +0000
5 weeks ago	Tue, 17 Dec 2013 17:58:05 +0000

11

11

## Kombiniranje date() i strtotime()

- problem: pronaći datum prvog petka u idućem mjesecu

```
$slijedecimj=date('Y-' . (date('n')+1) . '-01') ;
$slijedecimjTS=strtotime($slijedecimj);
$prvipetakTS=strtotime("Friday",$slijedecimjTS);
echo 'Danas je ' .date('d.m.Y.') . '<br>';
echo 'Prvi petak idućeg mjeseca je ' .
    date ('d.m.Y.',$prvipetakTS);
```

```
Danas je 21.01.2014.
Prvi petak idućeg mjeseca je 07.02.2014.
```

12

12

## Dohvaćanje dana i tjedna u godini

- ▶ dohvaćanje rednog broja dana u godini moguće je korištenjem malog slova z u prvom argumentu funkcije date()

```
$mojidatumi=array('2014-01-01','2014-01-21',
                   '2014-06-30','2014-12-31');
foreach($mojidatumi as $datum) {
    $zig=strtotime($datum);
    echo "Dan: " . date("d.m.Y. : z", $zig) . "<br />";
}
```

```
Dan: 01.01.2014. : 0
Dan: 21.01.2014. : 20
Dan: 30.06.2014. : 180
Dan: 31.12.2014. : 364
```

13

13

## Dohvaćanje dana i tjedna u godini

- ▶ dohvaćanje rednog broja tjedna u godini moguće je korištenjem W u prvom argumentu funkcije date()

```
$mojidatumi=array('2014-01-01','2014-01-21',
                   '2014-06-30','2014-12-31');
foreach($mojidatumi as $datum) {
    $zig=strtotime($datum);
    echo "Datum " . date("d.m.Y. = w", $zig) .
         ". tjedan<br />";
}
```

```
Datum 01.01.2014. = 01. tjedan
Datum 21.01.2014. = 04. tjedan
Datum 30.06.2014. = 27. tjedan
Datum 31.12.2014. = 01. tjedan
```

14

14

## Još par funkcija i funkcionalnosti...

- ▶ provjera je li godina prijestupna moguća je korištenjem L u prvom argumentu funkcije date()
- ▶ Funkcija vraća 1 ako je, a 0 ako nije prijestupna

```
$god=2014;  
$vrzig=strtotime("$god-01-01");  
if(date('L',$vrzig)==1) {  
    echo "Godina $god je prijestupna <br />";  
}  
else {  
    echo "Godina $god nije prijestupna <br />";  
}
```

Godina 2014 nije prijestupna

15

15

## Još par funkcija i funkcionalnosti...

- ▶ zanima nas izlazak i zalazak sunca?
  - date\_sunrise() i date\_sunset()

```
$zgDuz=16;  
$zgSir=45.8;  
$vrZona=1;  
$zenit=90;  
echo date("l d.m.Y."). ' izlazak sunca je u ' .  
    date_sunrise(time(), SUNFUNCS_RET_STRING, $zgSir,  
    $zgDuz, $zenit, $vrZona);
```

Tuesday 21.01.2014. izlazak sunca je u 07:32

16

16



# P11: PHP, JavaScript i AJAX

Kolegij: Programiranje u PHP  
Predavač: Dalibor Bužić

## JavaScript i PHP

- ▶ JavaScript je najpopularniji klijentski skriptni jezik
- ▶ JavaScript nema veze s programskim jezikom Java (kojog je web tek jedno od područja primjene)
- ▶ JScript je Microsoftova inačica JavaScripta
- ▶ izvršava se u klijentovom web pregledniku
- ▶ ima direktni pristup svim elementima web dokumenta
- ▶ PHP i JavaScript imaju dosta sličnosti u sintaksi
- ▶ možda najveća razlika PHP-a i JavaScripta je u objektnom modelu – konceptualna, ali i sintaksna (JavaScript koristi *dot* objektnu notaciju (objekt.postupak), a PHP *arrow* notaciju (\$objekt->postupak))
- ▶ temeljem ovih razlika pristup istom objektu iz JavaScripta i PHP-a nije moguć!

## JavaScript i PHP

- ▶ JavaScript ne može direktno komunicirati s bazom podataka
- ▶ PHP posjeduje veliku snagu u stvaranju dinamičkih web stranica, ali on je striktno poslužiteljski jezik
- ▶ čitav je niz zadaća na web stranicama koje ne zahtijevaju procesorsku snagu poslužitelja, a za te zadaće potrebna je brza reakcija (npr. promjena izgleda gumba kad se prelazi mišem preko njega)
- ▶ iako bi netko mogao očekivati da bi se korištenjem server-side PHP-a i client-side JavaScripta na istoj stranici mogli generirati brojni problemi, stvarnost je suprotna: međusobne različitosti ovih jezika čine ih dobrim spojem

3

3

## JavaScript i PHP

- ▶ **kad koristiti JavaScript?**
  - jednostavne aritmetičke operacije na formi i kalkulatori (računanje iznosa košarice ili kalkulator otplate kredita)
  - jednostavna provjera ispravnosti podataka unesenih na formu
  - navigacija stranicama (pull-down izbornici)
  - pop-upovi
  - događaji web preglednika (mouseove, onClick)
- ▶ **dvije stvari treba imati na umu:**
  - programer uvijek treba pretpostaviti da će korisnik unijeti neispravne podatke
  - JavaScript može biti onemogućen!

4

4

## JavaScript i PHP

- ▶ zato treba pristupiti na sljedeći način: prepostaviti da se JavaScript provjera podataka nije dogodila i uvijek provesti rigoroznu validaciju u PHP-u (neki zlonamjernik može poslati 'pokvarene' podatke serveru zaobilaženjem JavaScript provjera)
- ▶ jednom kad su podaci ušli u PHP, korisnik ih više ne može kontrolirati pa se smanjuje mogućnost za neželjene posljedice
- ▶ općenito pravilo (ali ne uvijek) je: provjera ispravnosti podataka koristi se i u JavaScriptu i u PHP-u.

5

### Primjer: FormaZaUnos.php

```
<html><head><title>Forma za unos</title>
<script>
function provjeri(form) {
greska = provjeriIme(form.ime.value)
greska += provjeriPrezime(form.prezime.value)
if (greska == "") return true
else { alert(greska); return false }
}
</script></head><body>
<table border="0" cellpadding="2" cellspacing="5"
bgcolor="#FCC66"> <th colspan="2" align="center">Unos novog
korisnika</th>
<form method="post" action="UnesiKorisnika.php"
onSubmit="return provjeri(this)">
<tr><td>Ime</td><td><input type="text" name="ime" /></td>
</tr><tr><td>Prezime</td><td><input type="text"
name="prezime" /></td>
</tr><tr><td colspan="2" align="center">
<input type="submit" value="Unesi" /></td>
</tr></form></table>
```

6

6

## Primjer: FormaZaUnos.php (...nastavak)

```
<script>
function provjeriIme(polje) {
if (polje == "") return "Ime nije uneseno.\n"
return ""
}
function provjeriPrezime(polje) {
if (polje == "") return "Prezime nije uneseno.\n"
return ""
}
</script></body></html>
```

The screenshot shows a simple web form titled "Unos novog korisnika". It contains two input fields: one for "Ime" and one for "Prezime", both with placeholder text. Below the inputs is a single "Unesi" button.

7

7

```
<?php
$ime=""; $prezime="";
if (isset($_POST['ime'])) $ime = $_POST['ime'];
if (isset($_POST['prezime'])) $prezime = $_POST['prezime'];
$greska = provjeri_ime($ime);
$greska .= provjeri_prezime($prezime);
echo "<html><head><title>Primjer</title>";
if ($greska == "") {
echo "</head><body>Podaci su uspješno uneseni: $ime,
$prezime.</body></html>";
exit;
}
echo <<<_PONOVNO
<script type="text/javascript">
function provjeri(form) {
greska = provjeriIme(form.ime.value)
greska += provjeriPrezime(form.prezime.value)
if (greska == "") return true
else { alert(greska); return false }
}
</script></head><body>
```

8

8

## Primjer: UnesiKorisnika.php (...nastavak)

```
<table border="0" cellpadding="2"
cellspacing="5" bgcolor="#FFCC66">
<th colspan="2" align="center">Unos novog korisnika</th>
<tr><td>Pronadene su sljedeće greške<br />
na formi: <p><font color=red>$greska</font></p></td></tr>
<form method="post" action="UnesiKorisnika.php"
onSubmit="return provjeri(this)">
<tr><td>Ime</td><td><input type="text" name="ime"
value="$ime" /></td>
</tr><tr><td>Prezime</td><td><input type="text"
name="prezime" value="$prezime" /></td>
</tr><tr><td colspan="2" align="center">
<input type="submit" value="Unesi" /></td>
</tr></form></table>
<script>
function provjeriIme(polje) {
if (polje == "") return "Ime nije uneseno.\n";
return "";
}
```

9

9

## Primjer: UnesiKorisnika.php (...nastavak)

```
function provjeriPrezime(polje) {
if (polje == "") return "Prezime nije uneseno.\n";
return "";
}
</script></body></html>
_PONOVNO;
function provjeri_ime($polje) {
if ($polje == "") return "Nije uneseno ime<br />";
return "";
}
function provjeri_prezime($polje) {
if ($polje == "") return "Nije uneseno prezime<br />";
return "";
}
?>
```

Unos novog korisnika  
Pronadene su sljedeće greške  
na formi:

Nije uneseno prezime

Ime	<input type="text" value="Antonija"/>
Prezime	<input type="text"/>
<input type="button" value="Unesi"/>	

10

10

## AJAX

- ▶ AJAX (Asynchronous JavaScript and XML)
- ▶ AJAX nije novi programski jezik, već novi način korištenja postojećih standarda
- ▶ to je tehnika za stvaranje brzih i dinamičnih web stranica
- ▶ omogućuje razmjenu podataka s poslužiteljem te ažuriranje dijelova web stranice bez ponovnog učitavanja cijele stranice
- ▶ razmjena podataka se odvija asinkrono s poslužiteljem
- ▶ osnova AJAX-a su internet standardi, odnosno kombinacija:
  - XMLHttpRequest objekta (za asinkronu razmjenu podataka sa serverom)
  - JavaScript/DOM (za prikaz/interakciju)
  - CSS
  - XML (često se koristi kao format za prijenos podataka)
- ▶ AJAX aplikacije su neovisne o pregledniku i platformi

11

11

## AJAX

### Preglednik

Dogodi se događaj...  
=> Stvara se objekt XMLHttpRequest  
=> Šalje se HTTPRequest



Poslužitelj  
=> Obrađuje XMLHttpRequest  
=> stvara odgovor i šalje podatke nazad pregledniku



12

12

## Primjer: AjaxDemo.html

```
<html><head><title>AJAX primjer</title>
</head><body><center />
<h1>Učitavanje web stranice</h1>
<div id='info'>Ova rečenica će biti zamijenjena</div>
<script>
param = "url=www.vsite.hr"
zahtjev = new ajaxZahtjev()
zahtjev.open("POST", "Server.php", true)
zahtjev.setRequestHeader("Content-type", "application/x-www-
form-urlencoded")
zahtjev.setRequestHeader("Content-length", param.length)
zahtjev.setRequestHeader("Connection", "close")
```

13

13

## Primjer: AjaxDemo.html (...nastavak)

```
zahtjev.onreadystatechange = function()
{ if (this.readyState == 4)
  { if (this.status == 200)
    { if (this.responseText != null)
      { document.getElementById('info').innerHTML =
        this.responseText
      }
      else alert("AJAX greška: nisu zaprimljeni podaci")
    }
    else alert( "AJAX greška: " + this.statusText)
  }
}
zahtjev.send(param)
```

14

14

## Primjer: AjaxDemo.html (...nastavak)

```
function ajaxZahtjev()
{
    try
    {
        var request = new XMLHttpRequest()
    }
    catch(e1)
    {
        try
        {
            request = new ActiveXObject("Msxml2.XMLHTTP")
        }
        catch(e2)
        {
            try
            {
                request = new ActiveXObject("Microsoft.XMLHTTP")
            }
            catch(e3)
            {
                request = false
            }
        }
    }
    return request
}
</script></body></html>
```

15

15

## Primjer: Server.php

```
<?php
if (isset($_POST['url'])) {
echo file_get_contents("http://".Procisti($_POST['url']));
}
function Procisti($tekst) {
$tekst = strip_tags($tekst);
$tekst = htmlentities($tekst);
return stripslashes($tekst);
}
?>
```

16

16



VsITe

# P12: XML, JSON, Grafika, PDF

Kolegij: Programiranje u PHP-u  
Predavač: Dalibor Bužić

1

XML – primjer datoteke

```
<?xml version="1.0" encoding="UTF-8"?>
<- <Hrvatska>
    - <DatumTermin>
        <Datum>17.01.2018</Datum>
        <Termin>15</Termin>
    </DatumTermin>
    - <Grad autom="0">
        <GradIme>Zagreb-Maksimir</GradIme>
        <Lat>45.822</Lat>
        <Lon>16.034</Lon>
        - <Podatci>
            <Temp> 6.1</Temp>
            <Vlaga>85</Vlaga>
            <Tlak>1001.1</Tlak>
            <TlakTend>+2.8</TlakTend>
            <VjetarSmjer>W</VjetarSmjer>
            <VjetarBrzina> 2.3</VjetarBrzina>
            <Vrijeme>potpuno oblačno</Vrijeme>
            <VrijemeZnak>6</VrijemeZnak>
        </Podatci>
    </Grad>
    - <Grad autom="1">
        <GradIme>Zagreb-Grič</GradIme>
        <Lat>45.814</Lat>
        <Lon>15.972</Lon>
        - <Podatci>
            <Temp> 5.9</Temp>
            <Vlaga>86</Vlaga>
            <Tlak>1001.5</Tlak>
            <TlakTend>+3.1</TlakTend>
            <VjetarSmjer>W</VjetarSmjer>
            <VjetarBrzina> 3.4</VjetarBrzina>
            <Vrijeme>slab vjetar</Vrijeme>
            <VrijemeZnak>-</VrijemeZnak>
        </Podatci>
    </Grad>
- <Grad autom="0">
```

4

4

## XML – primjer

```
$xml=simplexml_load_file("vrijeme.xml") or  
die("Greška: Ne mogu kreirati objekt");  
echo $xml->Grad[0]->GradIme . "<br>";  
echo $xml->Grad[0]->Podatci->Temp . " °C <br>";  
//echo 'Vrijeme: ' . $xml->Grad[0]->Podatci->Vrijeme  
. " <br>";  
  
foreach($xml->children() as $gradovi) {  
    echo $gradovi->GradIme . " | ";  
    echo $gradovi->Podatci->Temp . " | ";  
    echo $gradovi->Podatci->Vrijeme . "<br>";  
}
```

5

5

## JSON

- ▶ JSON (JavaScript Object Notation) je standardizirani format podataka koji omogućuje jednostavnu razmjenu podataka.
- ▶ Poput XML-a, tekstualno je zasnovan, pa ga mogu razumjeti i ljudi i računala.
  - Podaci u JSON formatu zauzimaju manje mesta od onih u XML-u.
- ▶ JSON se zasniva na dvije osnovne strukture:
  - objekt – skup parova ključ/vrijednost (**ključ:vrijednost**) – svaki objekt počinje s '{' a završava s '}', parovi ključ/vrijednost se međusobno odvajaju zarezima
  - niz – uređena lista vrijednosti – niz počinje s '[' a završava s ']' dok se vrijednosti odvajaju zarezom.
- ▶ U JSON-u ključevi su uvijek stringovi, a vrijednosti mogu biti stringovi, brojevi, true/false, null, objekti i nizovi.
- ▶ Stringovi moraju biti unutar dvostrukih navodnika, a mogu sadržavati escape znakove (\n, \t...).

6

6

## json\_encode()

```
$U2=array('vokal'=>"Bono", 'bas'=>"Adam",
'bubnjevi'=>"Larry",'gitara'=>"The Edge");
echo (json_encode($U2));
{"vokal":"Bono","bas":"Adam","bubnjevi":"Larry","gitara":"The Edge"}
```

```
$U2=array("Bono", "Adam", "Larry", "The Edge");
echo (json_encode($U2));
["Bono","Adam","Larry","The Edge"]
```

```
$U2=array("Bono", "Adam", "Larry", "The Edge");
echo (json_encode($U2, JSON_FORCE_OBJECT));
{"0":"Bono","1":"Adam","2":"Larry","3":"The Edge"}
```

- ▶ Indeksirani niz može se enkodirati i kao niz i kao objekt
- ▶ Asocijativni nizovi se enkodiraju samo kao objekti

7

7

## json\_decode()

```
$json=' {"vokal": "Bono", "bas": "Adam", "bubnjevi":
"Larry", "gitara": "The Edge"} ' ;
var_dump(json_decode($json)) ;
```

```
object(stdClass)#1 (4) { ["vokal"]=> string(4) "Bono" ["bas"]=> string(4) "Adam" ["bubnjevi"]=>
string(5) "Larry" ["gitara"]=> string(8) "The Edge" }
```

```
$json=' {"vokal": "Bono", "bas": "Adam", "bubnjevi":
"Larry", "gitara": "The Edge"} ' ;
var_dump(json_decode($json, true));
```

```
array(4) { ["vokal"]=> string(4) "Bono" ["bas"]=> string(4) "Adam" ["bubnjevi"]=>
string(5) "Larry" ["gitara"]=> string(8) "The Edge" }
```

8

8

## Pristup pojedinom elementu

```
$json='{"vokal": "Bono", "bas": "Adam", "bubnjevi": "Larry",  
"gitara": "The Edge"}' ;  
  
$sastavObj=json_decode($json);  
echo $sastavObj->vokal . '<br>';  
echo $sastavObj->bubnjevi . '<br>';  
  
$sastavNiz=json_decode($json, true);  
echo $sastavNiz['gitara'] . '<br>';  
echo $sastavNiz['bas'] . '<br>';
```

9

9

## Korištenje foreach()

```
$json='{"vokal": "Bono", "bas": "Adam", "bubnjevi": "Larry",  
"gitara": "The Edge"}' ;  
  
$sastavNiz=json_decode($json, true);  
foreach($sastavNiz as $kljuc=>$vrijednost){  
    echo $kljuc . "=>" . $vrijednost . "<br>";  
}  
  
$sastavObj=json_decode($json);  
foreach($sastavObj as $kljuc=>$vrijednost){  
    echo $kljuc . "=>" . $vrijednost . "<br>";  
}
```

vokal=>Bono  
bas=>Adam  
bubnjevi=>Larry  
gitara=>The Edge

10

10

## Grafika u PHP-u

- ▶ rad s grafikom omogućuje dodatak GD
  - podržava GIF, JPEG, PNG...
- ▶ sliku možemo učitati iz datoteke ili je stvoriti
- ▶ možemo je spremiti na disk ili prikazati u pregledniku
- ▶ `imagecreatetruecolor($sir, $vis)` – stvara novu *true color* crnu sliku danih dimenzija  
`$im = imagecreatetruecolor(300, 200);`
- ▶ `imagecreatefromjpeg()`, `imagecreatefromgif()`, `imagecreatefrompng()` – učitava sliku iz datoteke u memoriju  
`$im = imagecreatefrompng("logo.png");`

11

11

## Grafika u PHP-u

- ▶ prikaz slike odvija se u dva koraka:
  1. šalje se zaglavje Content-Type
  2. šalje se binarni sadržaj slike

```
header('Content-Type: image/png');  
imagepng($im);  
▶ ako dolazi do greške prilikom prikaza, onda treba  
staviti naredbu za slanje zaglavja kao prvu  
naredbu nakon otvarajuće oznake za PHP kod  
▶ kod spremanja slike nema potrebe slati zaglavje  
imagepng($im, 'slikal.png');
```

12

12

```

<?php
header ('Content-Type: image/png');
$im = imagecreatetruecolor(400, 300)
    or die('Ne mogu inicijalizirati sliku!');
//imagecolorallocate ($image, $red, $green, $blue)
$boja_teksta = imagecolorallocate($im, 100, 200, 200);
//imagestring($image, $font, $x_coord, $y_coord, $text,
$text_color)
//$font može biti od 1 do 5, veći broj = veći font
//no za $font postoje i druge opcije
imagestring($im, 5, 100, 50, 'Osnove izrade slika u
PHP-u', $boja_teksta);
imagepng($im);
imagedestroy($im);
//oslobadanje memorije
?>

```

Osnove izrade slika u PHP-u

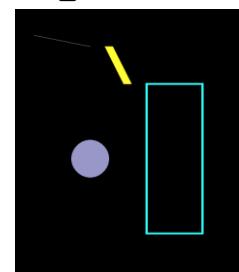
13

13

```

<?php
header ('Content-Type: image/png');
$im = imagecreatetruecolor(600, 800);
$boja1 = imagecolorallocate($im, 153, 150, 200);
$boja2 = imagecolorallocate($im, 255, 255, 51);
$boja3 = imagecolorallocate($im, 50, 255, 255);
// imageline($slika, $x1_Pocetak, $y1_Pocetak, // $x2_Kraj,
// $y2_Kraj, $boja)
imageline($im, 50, 70, 200, 100, $boja1);
// definiranje debljine u pikselima
imagesetthickness($im, 20);
imageline($im, 250, 100, 300, 200, $boja2);
imagesetthickness($im, 5);
// imagerectangle($slika, $x1_Dijag, $y1_Dijag,
// $x2_Dijag, $y2_Dijag, $boja)
imagerectangle($im, 350, 200, 500, 600, $boja3);
//imagefilledellipse($slika, $X_sredista, $Y_sredista, $sirina,
// $visina, $boja)
imagefilledellipse($im, 200, 400, 100, 100, $boja1);
imagepng($im);
imagedestroy($im); ?>

```



14

14

## Grafika u PHP-u

```
<?php
header ('Content-Type: image/png');
$im = imagecreatetruecolor(600, 800)
    or die('Ne mogu inicijalizirati sliku!');
$bojal = imagecolorallocate($im, 153, 150, 250);
putenv('GDFONTPATH=C:\WINDOWS\Fonts');
$font = getenv('GDFONTPATH') . '\comic.ttf';
//imagettftext($slika, $velicina, $kut, $x, $y, ...
imagettftext($im, 20, 0, 10, 50, $bojal, $font, "TrueType
fontovi");
$font = getenv('GDFONTPATH') . '\times.ttf';
imagettftext($im, 20, 0, 10, 90, $bojal, $font, "TrueType
fontovi");
$font = getenv('GDFONTPATH') . '\verdana.ttf';
imagettftext($im, 20, 0, 10, 130, $bojal, $font, "TrueType
fontovi");
imagettftext($im, 20, 270, 100, 150, $bojal, $font, "Okomiti
tekst");
imagepng($im);
imagedestroy($im);
?>
```

TrueType fontovi  
TrueType fontovi  
TrueType fontovi

Okomiti  
tekst

15

15

## Grafika u PHP-u - dinamičko iscrtavanje gumba...

```
<?php
header("Content-Type: image/png");
putenv('GDFONTPATH=C:\WINDOWS\Fonts');
$font = getenv('GDFONTPATH') . '\comic.ttf';
$velicina = isset($_GET['size']) ? $_GET['size'] : 12;
$text = isset($_GET['text']) ? $_GET['text'] : '';
$im = imagecreatefrompng("button.png");
$bojal = imagecolorallocate($im, 153, 0, 0);
if ($text) {
    // računanje pozicije teksta
    $tsize = imagettfbbox($velicina, 0, $font, $text);
    $dx = abs($tsize[2] - $tsize[0]);
    $dy = abs($tsize[5] - $tsize[3]);
    $x = (imagesx($im) - $dx) / 2;
    $y = (imagesy($im) - $dy) / 2 + $dy;
    // iscrtavanje teksta
    imagettftext($im, $velicina, 0, $x, $y, $bojal, $font, $text);
}
imagepng($im); //gumb se iscrtava ali je bez teksta
imagedestroy($im); ?>
```

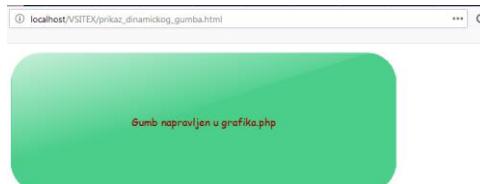
16

16

## ...dinamičko iscrtavanje gumba (nastavak)

- ▶ Skripta s prethodnog slajda može se pozvati s neke druge stranice:

```
<html>
    <head>
        <meta charset="UTF-8">
    </head>
    <body>
        
    </body>
</html>
```



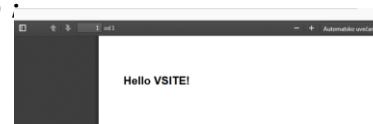
17

17

## PDF

- ▶ Zašto PDF?
  - ▶ za korištenje potrebno je uključiti neku od biblioteka, npr. FPDF, TCPDF...
  - ▶ FPDF se može preuzeti sa <http://www.fpdf.org/en/dl.php?v=16&f=zip>
  - ▶ raspakirati ga u odgovarajući direktorij
- "Fatal error: require(): Failed opening required 'fpdf.php'**  
**(include\_path='C:\xampp\php\PEAR') in C:\xampp..."**

```
ob_start(); //da ne šalje ispis odmah u preglednik
require('fpdf.php');
$pdf = new FPDF();
$pdf->addPage();
$pdf->setFont("Arial", 'B', 14);
$pdf->cell(50, 20, "Hello VSITE!");
$pdf->output();
ob_end_flush();
```

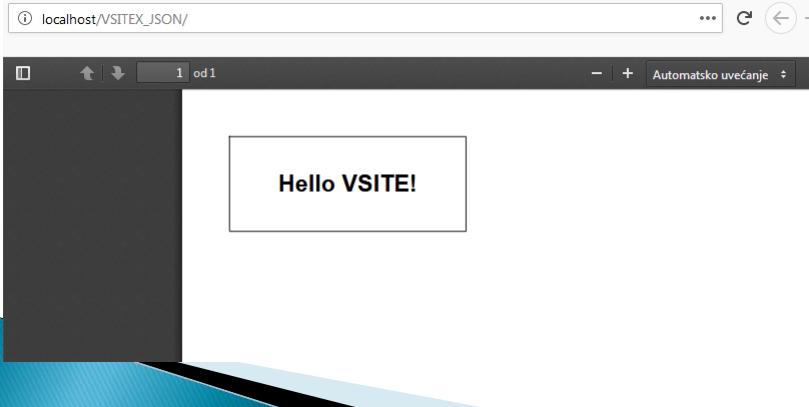


18

# PDF

- ▶ `Cell($sir, $vis, $text, $border, $ln, $align, $fill, $link)`

```
$pdf->cell(50, 20, "Hello VSITE!", 1, 0, 'C');
```



19

# PDF

```
ob_start();
require('fpdf.php');
$pdf = new FPDF('P', 'in', 'Letter');
//P=portret, in=inči, Letter=veličina stranice
$pdf->addPage();
$pdf->setFont('Arial', 'B', 24);
$pdf->cell(0, 0, "Gore lijevo!", 0,
$pdf->cell(6, 0.5, "Gore desno!", 1,
$pdf->ln(4.5); //koliko se jedinica
//pomičemo prema dolje
$pdf->cell(0, 0, "Ovo je sredina", 0
$pdf->ln(5.3);
$pdf->cell(0, 0, "Dolje lijevo!", 0,
$pdf->cell(0, 0, "Dolje desno!", 0,
$pdf->output();
ob_end_flush();
```



20

# PDF

```
ob_start();
require('fpdf.php');
$pdf = new FPDF();
$pdf->addPage();
$pdf->setFont("Arial", '', 14);
$pdf->cell(0, 5, "Normalni Arial tekst velicine 14", 0, 1,
'L');
$pdf->ln(); //pomak za visinu zadnje ispisane ćelije
$pdf->setFont("Arial", 'IBU', 18);
$pdf->cell(0, 15, "Ovo je ukoseni podebljani i potcrtani
tekst velicine 20", 0, 0, 'L');
$pdf->ln();
$pdf->setFont("Times", 'IU', 12);
$pdf->cell(0, 5, "Ovo je ukoseni potcrtani tekst velicine
15 Times", 0, 0, 'L');
$pdf->output();
ob_end_flush();
```

Normalni Arial tekst velicine 14

**Ovo je ukoseni podebljani i potcrtani tekst velicine 20**

*Ovo je ukoseni potcrtani tekst velicine 15 Times*

21

21

# PDF

```
ob_start();
require('fpdf.php');
$pdf = new FPDF();
$pdf->addPage();
$pdf->setFont("Times", 'U', 15);
$pdf->setTextColor(128);
$pdf->cell(0, 5, "Times font, potcrtani sivi tekst", 0, 0, 'L');
$pdf->ln();
$pdf->setTextColor(255, 0, 0);
$pdf->cell(0, 5, "Times font, potcrtani crveni tekst", 0, 0,
'L');
$pdf->output();
ob_flush();
```

Times font, potcrtani sivi tekst  
Times font, potcrtani crveni tekst

22

## PDF – primjer 1...

```
ob_start();
require('fpdf.php');
class MojPDF extends FPDF
{
function header()
{
global $naslov;
$this->setFont("Times", '', 12);
$this->setDrawColor(0, 0, 180);
$this->setFillColor(230, 0, 230);
$this->setTextColor(0, 0, 255);
$this->setLineWidth(1);
$sirina = $this->getStringWidth($naslov) + 150;
$this->cell($sirina, 9, $naslov, 1, 1, 'C', 1);
$this->ln(10);
}
function footer()
{
//Pozicija na 1.5 cm od dna
$this->setY(-15);
$this->setFont("Arial", 'I', 8);
$this->cell(0, 10,
"Ovo je podnozje stranice -> Stranica {$this->pageNo()}/{nb}", 0, 0, 'C');
}
}
```

23

23

## PDF – primjer 1 (nastavak)

```
$naslov = "Zaglavlje FPDF";

$pdf = new MojPDF('P', 'mm', 'Letter');
$pdf->aliasNbPages();
$pdf->addPage();
$pdf->setFont("Times", '', 24);
$pdf->cell(0, 0, "malo teksta na vrhu stranice", 0, 0,
'L');
$pdf->ln(225);
$pdf->cell(0, 0, "Jos teksta prema dnu...", 0, 0, 'C');
$pdf->addPage();
$pdf->setFont("Arial", 'B', 15);
$pdf->cell(0, 0, "Vrh stranice 2 nakon zaglavlja", 0, 1,
'C');
$pdf->output();
ob_flush();
```

24

24

## PDF – primjer 2

```
ob_start();
require('fpdf.php');
$pdf = new FPDF();
$pdf->addPage(); // prva stranica
$pdf->setFont("Times", '', 14);
$pdf->write(5, "Za link na sljedecoj stranici - kliknite ");
$pdf->setFont('', 'U');
$pdf->setTextColor(0, 0, 255);
$linkNaStr2 = $pdf->addLink();
$pdf->write(5, "ovdje", $linkNaStr2);
$pdf->setFont('');
$pdf->addPage(); // druga stranica
$pdf->setLink($linkNaStr2);
$pdf->image("C:/xampp/htdocs/VSITEX_JSON/logo.jpg", 10, 10, 30, 0, '',
"http://www.vsite.hr");
$pdf->ln(20);
$pdf->setTextColor(1);
$pdf->cell(0, 5, "Kliknite na sljedeci link ili na sliku", 0, 1, 'L');
$pdf->setFont('', 'U');
$pdf->setTextColor(0,0,255);
$pdf->write(5, "www.vsite.hr", "http://www.vsite.hr");
$pdf->output();
ob_flush();
```

25

25