Tehničko veleučilište u Zagrebu

Politehnički specijalistički diplomski stručni studij
specijalizacija Informatika

Napredne tehnike programiranja web servisa
(.open-source)

Petlje

- Petlje u PHP-u koristimo za izvršavanje iste akcije (grupe kodova) određeni broj puta.
- Ponekad želimo isti blok koda izvršiti nekoliko puta. U tom slučaju koristimo petlje na način korištenja petlje za izvršavanje te akcije.
- U PHP-u postoje slijedeće naredbe petlji:

Petlje

- **while** – prolazi blokom koda sve dok je specificirani uvjet istinit
- **do…while** - prolazi blokom koda jednom i nakon toga ponavlja petlju sve dok je specificirani uvjet istinit
- **for** …. prolazi blokom koda specificirani broj puta
- **foreach**…. prolazi blokom koda za svaki element u matrici

while naredba

- while naredba će izvršavati kod sve dok je uvjet istinit

```
while (uvjet)
kod koji se izvršava;
```

while naredba

- Slijedeći primjer pokazuje kako se petlja izvršava sve dok je varijabla manja ili jednaka 5. Svaki prolazak petlje povećava varijablu za 1.

```
<html>
<body>
  <?php
    $i=1;
      while($i<=5) {
          echo "Broj je " . $i . "<br />";
          $i++;
      }
  ?>
</body>
</html>
```

do … while naredba

- do…while naredba će izvršiti blok koda najmanje jednom a nakon toga će ponavljati petlju sve dok je uvjet istinit.

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

do... while naredba

- Vrijednost varijable $i će se povećati najmanje jednom i nastavit će se povećavati sve dok je manja od 5.

```
<html>
<body>
   <?php
      $i=0;
      do {
         $i++;
         echo "The number is " . $i . "<br />";
      }
      while ($i<5);
   ?>
</body>
</html>
```

for naredba

- for naredba se koristi ako znamo koliko puta želimo izvršiti naredbu ili niz naredbi.

```
for (initialization; condition; increment)
{
  kod koji će se izvršiti;
}
```

for naredba

- For naredba ima tri parametra:
  - prvi parametar inicira varijable,
  - drugi parametar uvjet,
  - treći parametar sadrži inkrement petlje.
- Ako je više od jedne varijable uključena u inicijalizaciju ili u prametar povećanja, oni trebaju biti odijeljeni zarezima. uvjet daje rezultat istina ili nije istina

for naredba

- Slijedeći primjer ispisuje text “Dobro jutro i dobar dan” pet puta:

```
<html>
<body>
   <?php
     for ($i=1; $i<=5; $i++) {
        echo "<p>Dobro jutro i dobar dan!</p>";
      }
   ?>
</body>
</html>
```

foreach naredba

- foreach naredba se koristi za prolaz kroz matrice.
- Za svaku petlju, vrijednost tekućeg elementa matrice je pridružena varijabli $value (pointer matrice je pomaknut za jedan), a u slijedećoj petlji promatramo slijedeći element.

```
foreach (array as value)
{
    code to be executed;
}
```

foreach naredba

- Slijedeći primjer pokazuje petlju koja ispisuje vrijednosti zadane matrice:

```
<html>
<body>
  <?php
    $arr=array("jedan", "dva", "tri");
      foreach ($arr as $value) {
        echo "<p>Vrijednost: " . $value . "</p>";
      }
?>
</body>
</html>
```

foreach naredba

- Prikaz na ekranu

```
Vrijednost: jedan
Vrijednost: dva
Vrijednost: tri
```

PHP include

- SSI – **server site includes** se koriste za kreiranje funkcija, zaglavlja, podnožja ili elemenata koji će se koristiti na više stranica.
- include() ili require() funkcijom možemo umetnuti sadržaj u PHP file prije nego se izvrši na serveru

Uključivanje drugih php datoteka

- Postoji mogućnost uključivanja drugih dokumenata u postojeći dokument
- Ako je kod koji pišemo prevelik možemo ga razdijeliti na cjeline koje ćemo kasnije po potrebi uključivati
- **Ovo se još naziva i modularnost koda**

Uključivanje drugih php datoteka

- Možemo imati aplikaciju podijeljenu u programske module koje po potrebi uključujemo u rad
- **Za to se koriste ugrađene funkcije u php-u:**
  - **require,**
  - **require_once,**
  - **include,**
  - **include_once**

require i include naredbe ili funkcije

- služe za uključivanje i evaluiranje php koda koji se nalazi u nekoj drugoj datoteci u trenutnu php skriptu
- require funkcija - ako se u uključenoj skripti dogodi greška, prekida se izvršavanje i uključene skripte i matične skripte
- include funkcija - prekida se izvršavanje samo uključene skripte ali ne i matične skripte

require i include naredbe ili funkcije

- Sintaksa

```
include "putanja do skripte";
ili
include('putanja do skripte');


require "putanja do skripte";
ili
require('putanja do skripte');
```

require i include naredbe ili funkcije

- Koji će biti rezultat?

```
<?php
      /* skripta_a.php */
      $a = 3;
?>
```

```
<html>
<body>
  <?php
       /* skripta.php */
       include( 'skripta_a.php');
       $b = $a * 2;
  ?>
</body>
</html>
```

require i include naredbe ili funkcije

- Uključivanje standardnog meni-a na sve stranice:

```
// menu.php
        <a href="http://www.xyz.com/default.php">Home</a> |
        <a href="http://www.xyz.com/about.php">About Us</a> |
        <a href="http://www.xyz.com/contact.php">Contact Us</a>
```

require i include naredbe ili funkcije

- Uključivanje standardnog meni-a na sve stranice:

```
<html>
<body>
    <?php
        php include("menu.php");
    ?>
        <h1>Welcome to my home page</h1>
        <p>Some text</p>
</body>
</html>
```

require i include naredbe ili funkcije

- U browseru bi source kod izgledao ovako:

```
<html>
<body>
	<a href="default.php">Home</a> |
	<a href="about.php">About Us</a> |
	<a href="contact.php">Contact Us</a>
	<h1>Welcome to my home page</h1>
	<p>Some text</p>
</body>
</html>
```

require i include naredbe ili funkcije

- Kada bi htjeli izmjeniti redoslijed veza ili dodati nove veze, samo bi promijenili dokument menu.php
- Stranica bi se automatski ažurirala
- require() funkcija je identična funkciji include() samo što je drugačije baratanje s greškama
- include() generira upozorenje, ali skripta se nastavlja izvršavati. Require() funkcija generira fatalnu grešku i zaustavlja se izvršavanje skripte

require i include naredbe ili funkcije

- Primjer include funkcije

```
<?php
include("wrongFile.php");
echo "Dobar dan!";
?>
```

```
Warning: include(wrongFile.php) [function.include]:
failed to open stream:
No such file or directory in C:\home\website\test.php on line 5
Warning: include() [function.include]:
Failed opening 'wrongFile.php' for inclusion
(include_path='.;C:\php5\pear')
in C:\home\website\test.php on line 5
Dobar dan!
```

require i include naredbe ili funkcije

- Primjer require funkcije

```
<?php
require("wrongFile.php");
echo "Dobar dan!";
?>
```

```
Warning: require(wrongFile.php) [function.require]:
failed to open stream:
No such file or directory in C:\home\website\test.php on line 5
Fatal error: require() [function.require]:
Failed opening required 'wrongFile.php'
(include_path='.;C:\php5\pear')
in C:\home\website\test.php on line 5
```

require_once i include_once naredbe ili funkcije

- **require_once** i **include_once** su identične prema **require** i **include**
- Kod se ne multiplicira nakon višestrukog ponavljanja require i include naredbi sa istim parametrima

require_once i include_once naredbe ili funkcije

- Sintaksa

```
<html>
<body>
   <?php
        include_once "putanja do skripte";
        ili
        include_once('putanja do skripte');

        require_once "putanja do skripte";
        ili
        require_once("putanja do skripte");
</body>
</html>
```

require_once i include_once naredbe ili funkcije

- Primjer

```
<?php
        /* skripta.php */
        $a++;
?>
```

```
<html>
<body>
<?php
/* skripta_a.php*/
$a = 0;
require 'skripta.php'; //uključujemo skriptu 1. put
require 'skripta.php'; //uključujemo skriptu 2. put
echo $a;
</body>
</html>
```

require_once i include_once naredbe ili funkcije

- Primjer

```
<html>
<body>
<?php
/* skripta_b.php */
$a = 0;
require_once 'skripta.php';
require_once 'skripta.php'; //ovaj put se skripta ne uključuje!
echo $a;
</body>
</html>
```

PHP funkcije

- Funkcija je skup naredbi koje se izvršavaju prema potrebi
- sve funckije počinju sa **function()**
- imena funkcija – preporučljiva su intiutivna imena a dozvoljeno je korištenje slova i _ nije dozvoljeno korištenje brojeva!
- kod funkcije počinje i završava zagradama – "{" i "}"

PHP funkcije

- Jednostavna funkcija koja ispisuje moje ime i prezime

```
<html>
<body>
   <?php
        function ispisiime()
        {
           echo "Moje ime i prezime";
        }
        ispisiime();
    ?>
</body>
</html>

// Izlaz: Moje ime i prezime
```

PHP funkcije

- Primjer

```
<html>
<body>
   <?php
         function ispisiime()  {
            echo "PHP 5.4.13";
         }
         echo "Dobar dan! ";
         echo "Moje ime je: ";
         ispisiime();
   ?>
</body>
</html>
```

```
// Izlaz: Dobar dan! Moje ime je: PHP 5.4.13
```

PHP funkcije

- Funckija **ispisiime** je vrlo jednostavna funkcija koja vraća statički string. Dodavanjem parametara možemo dodati puno više funkcionalnosti funkciji. Parametri su slični varijablama.
- Parametri se specificiraju unutar okruglih zagrada funkcije.

PHP funkcije

- Slijedeći primjer će ispisati različita imena i uvijek isto prezime

```
function ispisiime($fname) {
        echo $fname . " Horvat.<br />";
 }
      echo "Moje ime je ";
      ispisiime("Drago");
      echo "Moje ime je ";
      ispisiime("Hrvoje");
      echo "Moje ime je ";
      ispisiime("Stjepan");
```

```
Moje ime je Drago Horvat.
Moje ime je Hrvoje Horvat.
Moje ime je Stjepan Horvat.
```

PHP funkcije

- Funkcije se mogu koristiti za vraćanje vrijednosti

```
function add($x,$y)
 {
 $total = $x + $y;
 return $total;
 }
echo "1 + 16 = " . add(1,16)

// Izlaz: 1 + 16 = 17
```

- Ispis echo pod navodnicima ne radi operaciju zbrajanja, već samo ispis na ekran

PHP funkcije

- Primjer

```
function ducan($stanje="otvoren"){
echo "Ducan je $stanje";
}
// ispis default vrijednosti
ducan();
echo "<br />";

// ispis zadane vrijednosti
ducan("zatvoren");
```

```
Ducan je otvoren
Ducan je zatvoren
```

PHP funkcije

- date() funkcija specificira kako će se formatirati date/time.
- Koriste se slova za prezentiranje datuma i vremena. Npr.:
  - d – dan u mjesecu
  - m- tekući mjesec kao broj 1 -12
  - Y – tekuća godina u 4 znaka
- Znakovi kao što su "/", ".", or "-" mogu biti umetnuti između slova za dodatno formatiranje.

PHP funkcije

- Primjer

```
echo date("Y/m/d");
echo "<br />";
echo date("Y.m.d");
echo "<br />";
echo date("Y-m-d");
```

```
2013/04/08
2013.04.08
2013-04-08
```

PHP funkcije

- Primjer mktime

```
$sutra = mktime(0,0,0,date("m"),date("d")+1,date("Y"));
echo "Sutra je  ".date("Y/m/d/", $sutra);
```

```
Sutra je 2013/04/10
```

PHP funkcije

- Koliko je dana prošlo od početka godine

```
$days = floor((time()-mktime(null,null,null,1,0,date("Y")))/86400);

echo "$days dana je prošlo od početka godine";
```

```
99 dana je prošlo od početka godine
```

PHP funkcije

- Saznajte koliko ima dana u mjesecu u tekućoj godini

```
function days_in_month($year, $month) {
   return( date( "t", mktime( 0, 0, 0, $month, 1, $year) ) );
}
echo days_in_month(2013, 02);
```

```
// Izlaz: 28
```

```
function days_in_month($year, $month) {
   return( date( "t", mktime( 0, 0, 0, $month, 1, $year) ) );
}
echo days_in_month(2012, 02);
```

```
// Izlaz: 29
```

PHP funkcije

- PHP File Handling
- fopen() funkciju koristimo za otvaranje datoteka u PHP-u
- Prvi parametar funkcije sadrži ime datoteke koja će biti otvorena a drugi parametar specificira na koji način će biti otvorena datoteka.

```
<html><body>
   <?php
            $file=fopen("welcome.txt","r");
    ?>
</body></html>
```

PHP funkcije

- Datoteka može biti otvorena na slijedeće načine

Mode	Opis
r	otvara datoteku za čitanje, file pointer pokazuje na početak datoteke
r+	otvara datoteku za čitanje i pisanje, file pointer pokazuje na početak datoteke
w	otvara datoteku za pisanje, i briše dosadašnji sadržaj
w+	otvara datoteku za čitanje i pisanje, i briše dosadašnji sadržaj
a	otvara datoteku za pisanje, file pointer pokazuje na kraj datoteke
a+	otvara datoteku za pisanje, file pointer pokazuje na kraj datoteke
x	stvara novu datoteku, i otvara ju za pisanje, ako već postoji vraća false i error
x+	stvara novu datoteku, i otvara ju za čitanje i pisanje, ako već postoji vraća false i error
c	otvara datoteku za pisanje, file pointer pokazuje na početak datoteke
c+	otvara datoteku za čitanje i pisanje, file pointer pokazuje na početak datoteke

PHP funkcije

- Ako fopen() funkcija ne može otvoriti file vraća 0
- Primjer generira poruku ako fopen() funkcija ne može otvoriti specificirani file

```
<html>
<body>
   <?php
      $file=fopen("welcome.txt","r") or exit("Unable to open file!");
    ?>
</body>
</html>
```

PHP funkcije

- fclose() funkcija za zatvaranje otvorene datoteke

```
<html>
<body>
   <?php
      $file = fopen("test.txt","r");
      //some code to be executed
      fclose($file);
    ?>
</body>
</html>
```

PHP funkcije

- **Zatvarnje datoteke**
- fclose() funkcija za zatvaranje otvorene datoteke

```
<html>
<body>
   <?php
      $file = fopen("test.txt","r");
      //some code to be executed
      fclose($file);
    ?>
</body>
</html>
```

PHP funkcije

- **Kontrola završetka datoteke**
- feof() funkcija provjerava kraj datoteke odnosno da li je postignut "end-of-file" (EOF).
- feof() funkcija je korisna za formiranje petlje kroz podatke nepoznate dužine.
- Zapamtimo da ne možemo čitati iz datoteke otvorene na način w i x.

```
<?php
      if (feof($file)) echo "End of file";
    ?>
```

PHP funkcije

- **Čitanje datoteke line by line …fgets()**
- fgets() funkcija se koristi za čitanje pojedinačne linije datoteke
- Nakon poziva funkcije pointer se pomiće na slijedeču liniju
- Primjer čita datoteku liniju po liniju sve dok nije postignut EOF

```
$file = fopen("welcome.txt", "r") or exit("Unable to open file!");
//Output a line of the file until the end is reached
while(!feof($file)) {
    echo fgets($file). "<br />";
  }
fclose($file);
```

PHP funkcije

- **Čitanje datoteke znak po znak**
- fgetc() funkcija se koristi za čitanje pojedinačnog karaktera iz datoteke
- Nakon poziva te funkcije file pointer se pomiće do slijedećeg karaktera.
- U slijedećem primjeru pokazan je način čitanja datoteke, znak po znak dok se ne detektira EOF

PHP funkcije

- Primjer

```
<?php
$file=fopen("welcome.txt","r") or exit("Unable to open file!");
while (!feof($file)) {
      echo fgetc($file);
 }
fclose($file);
?>
```

Funkcije za rad sa sessionima i cookiesima

- U web aplikacijama često je korisno zapamtiti da li je korisnik već posjetio stranicu i zapamtiti neke njegove postavke. To postižemo sessionima i cookiesima.
- **Cookies** je mehanizam kojim se dodaje stanje u HTTP zahtjeve.
- Vrijednost se pohranjuju na klijentu.

Funkcije za rad sa sessionima i cookiesima

- Daljnjim parametrima je moguće postaviti vrijeme trajanja cookiea, put na serveru na kojem će cookie biti dostupan, domenu i stupanj sigurnosti.
- **$name = $_COOKIE["username"];**

Funkcije za rad sa sessionima i cookiesima

- Skripta provjerava da li je korisnik u formi upisao svoje korisničko ime, i ako je sprema taj podatak u cookie varijablu 'user', i postavlja njegovo trajanje na 24 sata

```
<?php
$user = $_POST['user'];
if( $user != null ){
      setcookie( "username", $user , time() + 86400 );
}
?>
```

Funkcije za rad sa sessionima i cookiesima

- Skripta provjerava da li je postavljenja cookie varijabla 'user', te ako je, njenu vrijednost sprema u varijablu i ispisuje na ekranu.

```
<?php
if (isset($_COOKIE['username'])) {
    $user = $_COOKIE['username'];
    echo "Hello ".$user."!<br/>";
}
?>
```

Funkcije za rad sa sessionima i cookiesima

- Cookies mehanizam ima i nekoliko mana:
  - preglednik ne mora prihvatiti cookie, a funkcija setcookie ne javlja da li ga je prihvatio.
  - problem sigurnosti,
  - ograničenja na količinu poslane informacije
- Umjesto cookiesa možemo koristiti session mehanizam, koji podatke pohranjuje na poslužitelju, a na klijentu se pohranjuje samo session ID kao cookie (ili kao GET parametar)

Funkcije za rad sa sessionima i cookiesima

- **session**
- Za korištenje varijable $_SESSION važno je prvo pozvati funkciju session_start()
- ***funkcija*** session_start();
- Funkcija inicijalizira superglobalno polje $_SESSION i vraća true ili false ovisno o tome da li je session uspješno pokrenut.
- Nakon što smo pozvali tu funkciju, možemo postavljati session varijable i pristupati im
- **$_SESSION[‘ime_varijable’]=vrijednost;**
- **$var=$_SESSION[‘ime_varijable’];**

Funkcije za rad sa sessionima i cookiesima

- Primjer

```
<?php
   session_start();
   if($_SESSION['login'] == ''){
      // Korisnik nije ulogiran
   } else {
      // Korisnik je ulogiran
   }
?>
```

Funkcije za rad sa sessionima i cookiesima

- Za provjeru podataka o korisniku potrebno je prvo izraditi HTML obrazac za prijavu korisnika

```
<html>
<head><title>Prijava korisnika</title></head>
<body>
Otvaranje novog korisničkog računa:<br>
    <form action='login.php' method='POST'>
      Korisnik: <input type='text' name='user'><br>
      Lozinka: <input type='password' name='pass'><br>
      <input type='submit' value='Pošalji'><br>
    </form>
</body>
</html>
```

Funkcije za rad sa sessionima i cookiesima

- Nakon unosa podataka podaci se šalju datoteci login.php koja ima sljedeći sadržaj

```
<?php
  // Provjera podataka
  session_start();
    if($_POST['user'] == 'asimec' && $_POST['pass'] == '1234'){
       $_SESSION['login'] = $_POST['user'];
    } else {
        unset($_SESSION['login']);
    }
  // Preusmjeravanje na naslovnu stranicu
  header('location: naslovnica.php');
?>
```

Funkcije za rad sa sessionima i cookiesima

- U provjeri podataka provjerava se da li je korisnik upisao ispravno korisničko ime (asimec) i lozinku (1234). Ako je postavlja se vrijednosti varijable $_SESSION['login'] na njegovo korisničko ime, a inače se briše.
- Nakon provjere koristi se funkcija header() koja preusmjerava preglednik na stranicu naslovnica.php.

Funkcije za rad sa sessionima i cookiesima

- Primjer naslovnica.php

```
<?php
  session_start();
  if(isset($_SESSION['login'])){
     echo „Dobrodošli korisniče „ . $_SESSION['login'] . „!<br>“;
     echo „Danas je: „. date(„l“). „<br>“;
   } else {
     echo „Upisali ste pogrešno korisničko ime ili lozinku!<br>“;
     echo „Pokušajte ponovo“;
  }
?>
```

Funkcije za rad sa sessionima i cookiesima

- Ako želimo provjeriti da li postoji neka session varijabla koristimo funkciju isset
- Za brisanje session podataka možemo koristiti session_unset i session_destroy.
- Funkcija session_unset oslobađa sve session varijable
- Funkcija session_destroy uništava sve podatke pridružene tekućem sessionu, ali ne oslobađa varijable, niti session cookie.

Funkcije za rad sa sessionima i cookiesima

- Primjer

```
<?php
  session_start();
  $_SESSION['username']=$_POST['username'];
?>
```

```
<?php
  session_start();
  if (isset($_SESSION['username'])) {
        echo "Username: ".$_SESSION[username']."<br/>";
   } else { echo "Unknown user, please log in!."<br/>"; }
?>
```

PITANJA?