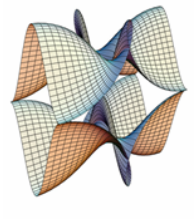

Prirodoslovno-matematički fakultet
Matematički odsjek
Sveučilište u Zagrebu

RAČUNARSKI PRAKTIKUM II

Predavanje 05 - PHP i Web programiranje

25. ožujka 2019.

Sastavio: Zvonimir Bujanović

Superglobalne varijable

Superglobalne varijable

- Skup ugrađenih varijabli koje su uvijek dostupne:
- `$_SERVER` - informacije o serveru i izvršnom okruženju
- `$_GET` - polje koje sadrži HTTP GET varijable
- `$_POST` - polje koje sadrži HTTP POST varijable
- `$_COOKIE` - polje koje sadrži HTTP Cookies
- `$_FILES` - polje koje omogućava *upload* datoteka preko HTTP-a
- `$_REQUEST` - unija polja `$_GET`, `$_POST` i `$_COOKIE`
- `$_SESSION` - polje koje sadrži varijable koje pripadaju *sesiji*
- `$_ENV` - varijable PHP "okruženja", za popis vidi `phpinfo();`

Ovo su sve asocijativna polja (`array`).
Pogledajte Primjer 1 uz ova predavanja.

SERVER

`$_SERVER` - informacije o serveru i izvršnom okruženju

- `'PHP_SELF'` – ime skripte koja se izvršava, bez imena servera. Npr. za http://example.com/foo/bar.php je /foo/bar.php.
- `'SERVER_ADDR'` – IP-adresa servera.
- `'SERVER_NAME'` – ime (virtualnog) hosta
- `'REQUEST_METHOD'` – GET/POST/HEAD/PUT
- `'DOCUMENT_ROOT'` – lokacija u serverovom datotečnom sustavu gdje je smješten web-site
- `'REMOTE_ADDR'`, `'REMOTE_HOST'` – IP-adresa i ime hosta za klijenta
- `'REQUEST_URI'` – URI koji je poslužio za pristup stranici

```
1 print_r( $_SERVER );
```

GET i POST

GET zahtjevi:

- Parametri su vidljivi iz URL, kroz tzv. *query string*:
  http://www.example.com?ime=Pero&starost=20
- Tada iz PHP-a:
  ```
  $_GET['ime']==='Pero'
  $_GET['starost']==='20' // Pazi: '20' je string!
  ```
- Mogu biti spremljeni u cache, ostaju u povijesti browsanja, mogu se *bookmarkirati*.
- Imaju ograničenje na duljinu parametara.
- Nikad ih se ne koristi za osjetljive podatke (passwordi i slično).
- Trebaju se koristiti se samo za dohvaćenje podataka sa servera.

GET i POST

POST zahtjevi:

- Parametri nisu vidljivi iz URL.
- Tipično se koristi za slanje podataka unesenih u formu.
- Iz PHP-a:
  ```
  $_POST['ime']==='Pero'
  $_POST['starost']==='20' // Pazi: '20' je string!
  ```
- Osjetljivi podaci su i dalje bez enkripcije ⇝ HTTPS.
- Nikad se ne spremaju u cache, ne ostaju u povijesti browsanja, ne mogu se *bookmarkirati*.
- Nema ograničenja na dužinu podataka koji se šalju.
- Mogu se koristiti i za dohvaćanje i za spremanje podataka na server.

GET i POST

GET:

- Moguće je lako sagraditi *query-string* iz PHP-a:

```php
$vars = array( 'ime' => 'Pero' ,
               'starost' => '20' );
$query_string = http_build_query( $vars );
$url = '/skripta.php?' . $query_string;
```

Sigurnosni aspekti:

- Uvijek provjeriti je li varijabla definirana:

```php
if( !isset( $_GET['ime'] ) ) {
    exit( 'Trebate unijeti ime.' );
}
```

- Za podatke dobivene od korisnika uvijek treba provesti:
  - sanitizaciju - ukloniti "zabranjene" znakove iz podataka.
  - validaciju - provjeriti jesu li podaci u ispravnom obliku.

GET i POST

Pristup podacima iz forme:

- Ako formi imamo: `<input name="nesto">`,
  onda u skripti postoji `$_GET["nesto"]` ili `$_POST["nesto"]`.
- Specijalno za **checkbox**, **name** treba završavati sa `[]`.

```
1  <form method="post" action="skripta.php">
2      Meals:<br>
3      <input type="checkbox" name="meals[]" value="breakfast">Breakfast<br>
4      <input type="checkbox" name="meals[]" value="lunch">Lunch<br>
5      <input type="checkbox" name="meals[]" value="dinner">Dinner<br>
6      <input type="submit" name="btn1" value="Submit Button 1">
7      <input type="submit" name="btn2" value="Submit Button 2">
8  </form>
```

- Tada je `$_POST["meals"]` polje koje sadrži odabrane vrijednosti.
- Uočite da će postojati i `$_POST["btn1"]` ili `$_POST["btn2"]`,
  ovisno o tome na koji je gumb korisnik kliknuo.

COOKIE

- Uređeni parovi (ključ, vrijednost) koje će web-site moći dohvatiti prilikom narednih pristupa site-u.
- Postavljanje novog cookie:
  `setcookie( 'boja', 'crvena', 1417608000 );`
- Treći argument je *epoch timestamp* kad cookie ističe; ako ga ne postavimo ističe kad se browser zatvori
  $\rightsquigarrow$ postaviti na `time()+x`, gdje je `x` broj sekundi do isteka.
- Cookie se mora postaviti prije ikakvog ispisa HTML-a!
- Čitanje postojećih cookie-a (nakon *reload-a* stranice):

```
1 if( isset( $_COOKIE['boja'] ) )
2     echo 'Tvoja boja je ' . $_COOKIE['boja'];
```

- Brisanje ranije definiranih cookie-a:
  `setcookie( 'boja' , '' , 1 );`
- Cookie (po defaultu) dijele sve skripte iz istog direktorija.

Zadatak 1

Napišite PHP skriptu `zadatak1.php` koja generira web-stranicu ovako:

- Web-stranica ima inicijalno bijelu boju pozadine.
- Ako je korisnik već ranije bio posjetio stranicu i promijenio joj boju pozadine, onda se prikazuje ta ranije odabrana boja pozadine.
- Na web-stranici se nalazi padajući izbornik (select) na kojem korisnik može odabrati nekoliko unaprijed zadanih boja. (Postavite neki odabir kao defaultan.)
- Također, nalazi se textbox u kojeg korisnik može unijeti HTML kod boje.
- Klikom su gumb submit, ponovno se poziva ista skripta koja:
  - Ako je išta upisano u textbox, postavlja tu boju pozadine na web-stranicu.
  - U protivnom, postavlja boju pozadine koja je odabrana u padajućem izborniku.

Možete li svojim unosom u formu ostvariti neplanirani efekt?

Neki sigurnosni aspekti

- Izuzetno je lako slučajno napraviti grešku koja će ugroziti sigurnost servera!
- Cross-site scripting (XSS)
  - Ubacivanje maliciozne skripte koja se izvršava na klijentu.
- Primjer: neka u `test.php` piše:

```
1 <p>Hello, <?php echo $_GET['username']; ?></p>
```

Što će se dogoditi ako netko pristupi adresi `test.php?name=Pero<script>alert('XSS');</script>`

- Treba provesti *sanitizaciju* svih podataka koje nam korisnik može poslati:

```
1 <p>Hello, <?php
2   echo htmlentities( $_GET['username'], ENT_QUOTES );
3 ?></p>
```

Neki sigurnosni aspekti

- Osim sanitizacije, treba provesti i *validaciju*, tj. provjeru jesu li podaci koje je poslao korisnik u dobrom formatu.
- Validaciju možemo provesti pomoću regularnih izraza:

```
1 if( !preg_match( '/^-?\d+$/', $_POST['saldo'] ) )
2     echo 'Saldo mora biti cijeli broj.';
```

- PHP podržava (i) regularne izraze tipa PCRE – iste kao na Programiranju 1.
- Funkciji `preg_match`:
  - Kao prvi parametar šaljemo regularni izraz kojeg želimo prepoznati.
  - Prvi i zadnji znak (gore: `/`) su "delimiteri".
  - Radi i s Unicode-om: tada reg. izraz ima oblik `'/.../u'`.
- Postoje i brojne druge funkcije za rad sa reg. izrazima.

Neki sigurnosni aspekti

- Za sanitizaciju i validaciju mogu poslužiti i PHP filtri.

```php
$email = "john.doe@example.com";

// Ukloni sve ilegalne znakove iz email adrese.
$email = filter_var( $email, FILTER_SANITIZE_EMAIL );

// Validiraj e-mail
if( filter_var( $email, FILTER_VALIDATE_EMAIL ) === false )
    echo "$email nije ispravna email adresa.";
else
    echo "$email je email adresa u ispravnom formatu";
```

Zadatak 2

Korigirajte kod Zadatka 1 tako da napravite sanitizaciju i validaciju podataka koje skripta dobiva od korisnika:

- Kao unos u textbox, dozvolite samo znak # iza kojeg slijedi tro- ili šesteroznamenkasti hex kod boje.
- Kao unos u select, dozvolite samo riječi sastavljene od max 20 slova engleske abecede.

Nikad ne vjerujte unosu korisnika!

Primjer 2: Autentifikacija korisnika sa COOKIE

- Forma za login ili podaci za ulogiranog korisnika:

```
1  if( isset( $username ) ) {
2    // Ako je korisnik ulogiran, ispiši poruku i gumb za logout.
3    echo "Dobro došao, $username.<br />"; ?>
4    <form method="POST" action="2 - Login.php">
5      <input type="hidden" name="logout">
6      <input type="submit" value="Log Out">
7    </form>
8    <?php
9  }
10 else {
11   // Ako nije ulogiran, ispiši formu za logiranje. ?>
12   <form method="POST" action="2 - Login.php">
13     Username: <input type="text" name="username"> <br />
14     Password: <input type="password" name="password"> <br />
15     <input type="submit" value="Log In">
16   </form>
17   <?php
18 }
```

Primjer 2: Autentifikacija korisnika sa COOKIE

- Funkcija za provjeru passworda:

```php
function validate( $user, $pass ) {
  $users = array(
    'pero' => 'perinasifra' ,
    'ana'  => 'aninasifra' );

  if( isset( $users[$user] ) && ( $users[$user] === $pass ) )
    return true;
  else
    return false;
}
```

- Postavljanje cookie-ja ako je uspio login:

```php
$secret_word = 'racunarski praktikum 2!!!';
if( isset( $_POST['username'] ) && isset( $_POST['password'] )
  && validate( $_POST['username'], $_POST['password'] ) )
  setcookie( 'login', $_POST['username'] . ',' .
    md5( $_POST['username' ] . $secret_word ) );
```

Primjer 2: Autentifikacija korisnika sa COOKIE

- Provjera je li korisnik već ulogiran, dohvaćanje usernamea:

```php
unset( $username );
if( isset( $_COOKIE['login'] ) ) {
  list( $c_username, $cookie_hash )
    = explode( ',' , $_COOKIE['login'] );
  if( md5( $c_username . $secret_word ) == $cookie_hash )
    $username = $c_username;
  else
    echo "Poslan je pokvareni kolačić :)" ;
}
```

- Brisanje cookie-a ako se korisnik odlogirao:

```php
if( isset( $username ) && isset( $_POST['logout'] ) ) {
  setcookie( 'login', '', 1 );
  unset( $username );
}
```

Session

"Sesije"

- Ponovnim učitavanjem stranice ili učitavanjem neke druge stranice na složenijem web-siteu se gube podaci koje je korisnik unio (osim ako ne postavimo cookie).
- "Sesije" služe za očuvanje "stanja" web-aplikacije tijekom korisnikovog korištenja.
- Korisniku se automatski pridružuje tzv. session id (obično putem cookie-a).
- Sesiju nastavljamo (ili započinjemo ako već ne postoji) sa `session_start();`
- Ovo treba pozvati na početku svake PHP-datoteke koja treba interakciju s istim korisnikom.
- Nakon toga možemo čitati, dodavati i brisati elemente u `$_SESSION`.
- Elementi od `$_SESSION` mogu biti bilo kojeg tipa, pa i objekti!
- Da se sesija prekine (npr. logout), treba: `session_unset(); session_destroy();`

Primjer 3: Autentifikacija korisnika sa session

- Na početak php-filea ide `session_start();`
- Forma i funkcija za provjeru passworda su iste kao prije.
- Postavljanje `$_SESSION['login']` ako je uspio login:

```
1 $secret_word = 'racunarski praktikum 2!!!';
2 if( isset( $_POST['username'] )
3     && isset( $_POST['password'] )
4     && validate( $_POST['username'], $_POST['password'] ) )
5 {
6     $_SESSION['login'] =
7         $_POST['username'] . ',' .
8         md5( $_POST['username' ] . $secret_word );
9 }
```

Primjer 3: Autentifikacija korisnika sa session

- Provjera je li korisnik već ulogiran, dohvaćanje usernamea:

```php
unset( $username );
if( isset( $_SESSION['login'] ) ) {
  list( $c_username, $cookie_hash )
    = explode( ',' , $_SESSION['login'] );
  if( md5( $c_username . $secret_word ) == $cookie_hash )
    $username = $c_username;
  else
    echo "Netko se igra sa sessionom :)" ;
}
```

- Zaustavljenje sesije ako se korisnik odlogirao:

```php
if( isset( $username ) && isset( $_POST['logout'] ) ) {
  unset( $username );
  session_unset();
  session_destroy();
}
```

Zadatak 3

Napišite skripte `zadatak3_index.php` i `zadatak3_pogodi.php` koje omogućavaju igranje igre "Pogađanje brojeva".

- Kad korisnik prvi put dođe na stranicu `zadatak3_index.php`, skripta ga pita za ime, te generira slučajan broj X između 1 i 100. (Naravno, ne ispiše ga korisniku.) Skripta šalje dobivene podatke skripti `zadatak3_pogodi.php`.
- Skripta `zadatak3_pogodi.php` daje korisniku priliku da proba pogoditi broj. Klikom na gumb "Pogodi", broj se šalje istoj skripti koja onda ispiše je li korisnikov broj veći, manji ili jednak X.
- Nakon što se broj pogodi, ispisuje se prigodna čestitka.

DZ:

- Riješite zadatak pomoću samo jedne skripte.
- Da igraču bude lakše, ispisujte i njegov najveći pokušaj koji je manji od X, kao i najmanji pokušaj koji je veći od X.

FILES

`$_FILES`

- Ako je forma sadržavala input tipa `file` imena `filename`, onda će postojati polje `$_FILES['filename']`.
- Ključevi u tom polju su:
  - `name` - ime uploadane datoteke kako ga je proslijedio browser.
  - `type` - tzv. MIME-type datoteke.
  - `size` - veličina datoteke u byteovima.
  - `tmp_name` - lokacija na koju je datoteka privremeno spremljena na serveru.
  - `error` - eventualna greška (`UPLOAD_ERR_OK` ako je sve OK).
- HTML-forma mora imati atribut `enctype="multipart/form-data"`.
  Defaultni enctype je `application/x-www-form-urlencoded`.
- Treba osigurati da PHP ima pravo pisanja u direktorij u kojeg ćemo spremiti file. Po defaultu Apache ima username i grupu `www-data`.
- Nikada ne spremati datoteku u direktorij dostupan preko web-a!

Primjer 4: Upload datoteke

Forma za upload datoteke:

```
1  if( $_SERVER['REQUEST_METHOD' ] == 'GET' ) { ?>
2    <form method="post"
3          action="fileUpload.php"
4          enctype="multipart/form-data">
5      <input type="file" name="document" /><br />
6      <input type="submit" value="Pošalji datoteku" />
7  </form>
8  <?php
9  }
10 else ...
```

Primjer 4: Upload datoteke

Premještanje datoteke na konačnu lokaciju:

```
... else {
  if( isset( $_FILES['document'] )
      && ( $_FILES['document']['error'] == UPLOAD_ERR_OK ) )
  {
    $newPath = '/tmp/' . basename( $_FILES['document']['name'] );
    if( $_FILES['document']['size'] > 2000 )
      echo 'Nije dozvoljeno slati datoteke veće od 2kB.';
    else if( move_uploaded_file(
              $_FILES['document']['tmp_name'], $newPath ) )
      echo "Datoteka je spremljena u $newPath";
    else
      echo "Ne mogu spremiti dateku u $newPath";
  }
  else
    echo "Nije poslan dobar file.";
}
```

"Nestandardne" HTTP poruke

- Tipično, PHP skripta prima zahtjeve GET tipa.
- Ako se obrađuju POST zahtjevi nastali iz HTML forme, onda su uobičajena dva tipa sadržaja navedena u zaglavlju HTTP poruke:
  - `Content-Type: application/x-www-form-urlencoded`
  - `Content-Type: multipart/form-data`
- Samo u gornjim slučajevima automatski se napune varijable `$_GET` i `$_POST`.
- PHP skripta može obrađivati i bilo koji drugi tip HTTP poruke. Do tipa poruke možemo doći ovako:

```php
$headers = getallheaders();
echo $headers['Content-Type'];
```

- Sadržaj nestandardnih HTTP zahtjeva u string spremamo ovako:

```php
$str = file_get_contents( 'php://input' );
```

"Nestandardne" HTTP poruke

- Tipično, PHP skripta vraća sadržaj tipa **`text/html`**.
- Moguće je promijeniti tip sadržaja koji se šalje, pozivom funkcije **`header`**.
- Tako npr. možemo korisniku poslati PDF datoteku:

```
1 header('Content-Type: application/pdf');
2 header('Content-Disposition: attachment; filename="zz.pdf"');
3 readfile('/documents/original.pdf');
```

- Možemo i preusmjeriti korisnika na neku drugu adresu:

```
1 header( 'Location: http://www.example.com/' );
2 exit();
```

- Pozivi funkcije **`header`** trebaju doći prije bilo kakvog ispisa (sa **`echo`** ili na drugi način).

Slanje i čitanje mail-a iz PHP-a

- Slanje maila

```php
$to =      'mirko@example.com';
$subject = 'Fantastična nova obavijest!' ;
$body =    'Iz PHP-a se može slati mail!';
$header =  "Reply-To: webmaster@example.com\r\n"
         . "Organization: Mirko Inc." ;
mail( $to, $subject, $body, $header );
```

- Čitanje maila

```php
$mail = imap_open( '{mail.server.com:143}',
                   'username', 'password' );
// Dohvati listu svih headera
$headers = imap_headers( $mail );
// Dohvati header za zadnju poruku
$last = imap_num_msg( $mail );
$header = imap_header( $mail, $last );
// Dohvati tijelo zadnje poruke
$body = imap_body( $mail, $last );
imap_close( $mail );
```

Neki sigurnosni aspekti

- Form spoofing / Cross-site request forgery
  - Korisnik je ulogiran na naš site preko session/cookie, te nam (tj. našoj skripti) je poslao neke podatke u formi.
  - Bez prekida sesije, korisnik ode na maliciozni site koji mu također ponudi naizgled "normalnu" formu.
  - No "normalna" forma zapravo šalje podatke našoj skripti na našem site-u.
  - Kako korisnik i dalje ima sesiju s našim siteom, skripta na našem site-u ne bi mogla razlikovati podatke koji joj stižu od forme na našem site-u i od forme na malicioznom.

Neki sigurnosni aspekti

- Rješenje: skriveno polje s random vrijednosti u formi.
- form.php

```php
<?php
  session_start();
  $_SESSION['token'] = md5( uniqid( mt_rand(), true ) );
?>
<form action="buy.php" method="POST">
  <input type="hidden" name="token"
         value="<?php echo $_SESSION['token']; ?>" />
  <input type="submit" value="Kupi!" />
</form>
```

- buy.php

```php
session_start();
if( !isset( $_SESSION['token'] ) ||
    $_POST['token'] != $_SESSION['token'] )
  echo 'Unesi username i password.';
else
  echo 'Sve OK, nastavljamo dalje...';
```

Neki sigurnosni aspekti

- Datoteke koje šalje korisnik uvijek treba smjestiti u direktorij nedostupan kroz web, te po mogućnosti preimenovati.
  - Ako korisnik može direktno pristupiti datoteci koju je poslao, mogao bi poslati npr. PHP skriptu i izvršavati proizvoljni kod na našem serveru.
- Korisničke šifre se uvijek čuvaju enkriptirane.
  - Enkripcija šifre (PHP 5.5 i noviji):

```php
$hash = password_hash( $_POST['password' ],
                       PASSWORD_DEFAULT );
// Sad spremi $hash u bazu podataka/datoteku
```

  - Provjera šifre prilikom logiranja (PHP 5.5 i noviji):

```php
// Učitaj $hash iz baze podataka/datoteke
if( password_verify( $_POST['password' ], $hash ) )
    echo "Login je uspio!";
else
    echo "Login nije uspio!";
```

  - U PHP 5.4 i ranije, kombinacija `md5` i skrivene riječi.

Zadatak 4

Napišite PHP skriptu **`zadatak4.php`** koja omogućava igranje igre križić-kružić.

- Pretpostavite da oba igrača igraju naizmjenično za istim računalom.
  (Zasad ne znamo implementirati da sjede svaki za svojim!)
- Uputa: ako u formi postoje buttoni tipa **`submit`** koji imaju nameove A i B, onda će biti definirano ili **`$_POST['A']`** ili **`$_POST['B']`**, ovisno o tome koji je gumb kliknut.
- Alternativno, svaka od 9 "kućica" za igru može sadržavati svoju formu, a u svakoj formi se uz vidljivi gumb nalazi i skriveno polje (**`input type="hidden"`**) koje sadrži identifikator kliknutog gumba.

Zadatak 5

Napišite PHP skriptu `zadatak5.php` koja sanitizira i validira, a zatim i ispisuje podatke poslane od strane forme iz Predavanja 2, Zadatak 1.