Obrada podataka poslanih preko web formi

Kreiranje forme na web stranici

Forme omogućuju komunikaciju korisnika i sustava (web stranice ili aplikacije). Sadrže HTML elemente za upis ili odabir (polja za upis teksta, padajući izbornici, liste za odabir itd.) podataka.

<form id="contact_form" action="" method="POST”>

Ime:

<input type=„text" name=„ime” value="" />

Prezime:

<input type=„text" name=„prezime” value="" />

Poruka:

<textarea name=„poruka”></textarea>

<input type="submit" name=„btn” value=„Posalji" />

</form>

Ova forma sadrži dva manja tekstualna polja i jedno veliko te gumb za slanje na daljnju obradu. HTML kod forme izvršava se u web pregledniku i korisniku se prikazuje forma.

Primjer padajućeg izbornika

Grad:

```
<select name=„grad”>
  <option value=„0">Odaberite:</option>
  <option value=„1">Zagreb</option>
  <option value=„2">Zadar</option>
  <option value=„3">Osijek</option>
</select>
```

Ostali elementi mogu biti checkbox i radio button (za mogućnost jednostrukog ili višestrukog odabira.

Nakon što korisnik popuni formu i klikne na gumb za slanje, podaci se šalju na obradu.

Prosljeđivanje podataka iz forme na obradu

Važni atributi su method i action. O njihovim vrijednostima ovisi pristup prikupljanju podataka: gdje će se točno proslijediti podatke i na koji način.

Atribut action predstavlja URL stranice na koju će se proslijediti podaci iz forme, ali može sadržavati i samo naziv skripte koja radi obradu: action=„obrada.php” ili action=„http://www.php.net/obrada.php”

Ako je atribut action prazan, preglednik će ponovno pozvati stranicu ili skriptu na kojoj se trenutno nalazimo.

Atribut method govori pregledniku na koji način treba proslijediti podatke: GET ili POST metodom, npr. method=„GET”

Varijabla $_GET

Ako je u formi definirano slanje podataka putem metode GET, za obradu podataka potrebna je posebna super-globalna varijabla $_GET dostupna u svim dijelovima programskog koda. Ona je polje podataka čiji ključevi predstavljaju vrijednosti elemenata atributa name poslanih s forme, dok vrijednosti elemenata polja odgovaraju vrijednostima poslanih elemenata.

Podaci su nakon slanja vidljivi u URL-u stranice na koju smo preusmjereni.

Nakon naziva skripte dolazi znak ? te vrijednost atributa name prvog elementa s poslane forme, pa znak = te vrijednost koju je korisnik upisao. Iza toga slijede ostali parovi tipa ključ-vrijednost poslani s forme, ali odvojeni znakom &.

Preuzimanje poslanih podataka: $ime=$_GET[„ime”]; $prezime=$_GET[„prezime”];

Upotreba metode GET

Metoda GET rijetko se rabi za slanje podataka s forme. Češća je kod poveznica.

Npr. ako na stranici uz sliku nekog proizvoda stoji poveznica:

Detalji

Klikom na poveznicu otvara se navedena stranica. Nakon znaka upitnika nalaze se vrijednosti koje se pohranjuju u varijablu $_GET. Daljnja obrada može npr. biti:

```
if($_GET[„task”]==„view”)
{
$id=$_GET[„ID”];
//.....
}
```

```
switch($_GET[„task”])
{
case ‘view’:
        //.....
```

Varijabla $_POST

Super globalna varijabla (polje podataka) u koju se pohranjuju podaci iz forme ako je definirana metoda slanja podataka POST.

```
if (isset($_POST[„btn”]))
{
echo ‘<pre>’;
print_r($_POST);
echo ‘</pre>’;
}
```

- provjerava se postoji li u varijabli $_POST element s ključem btn, odnosno da li je pritisnut.
- ugrađena PHP funkcija print_r služi za ispis sadržaja polja

Prikupljanje podataka pomoću varijable $_POST

$ime=$_POST[„ime”]; $prezime=$_POST[„prezime”];

Grupiranje podataka u formi promjenom atributa name za neke elemente u formi:

<form action="" method="POST”>

Ime:

<input type=„text" name=„korisnik[ime]” value="" />

Prezime:

<input type=„text" name=„korisnik[prezime]” value="" />

</form>

Unutar polja se nalazi novo polje pa je varijabla $_POST postala višedimenzionalno polje.

$ime=$_POST[„korisnik”][„ime”];

a može se koristiti i foreach za dohvat: foreach($_POST[„korisnik”] as $key=>$val) $$key=$val; echo $ime.’ ‘ .$prezime;

Ovdje petlja prolazi kroz polje unutar elementa korisnik, razdvaja podatke na ključ i vrijednost te pomoću $$ dinamički stvara nove varijable $ime i $prezime.

Varijabla $_REQUEST

Super-globalna varijabla koja sadrži vrijednosti više super-globalnih varijabli. Korisna kad nismo sigurni kako će do naše skripte podaci doći (GET ili POST).

$ime=$_REQUEST[„ime”];

$prezime=$_REQUEST[„prezime”];

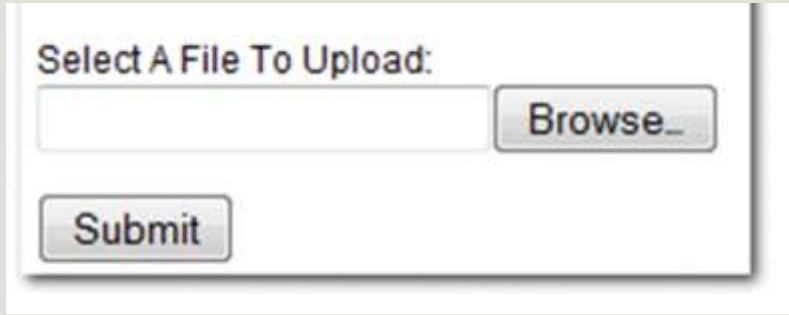
$poruka=$_REQUEST[„poruka”];

Slanje datoteka preko web formi

Preko formi moguće je slanje datoteka (upload) na poslužitelj (često u aplikacijama i prijavama). Korisnik datoteku sa svog računala može pohraniti na poslužitelj, a PHP nudi mehanizme za rad sa datotekama.

Priprema forme za prijenos datoteka:

```
<form method=„POST” action=„” enctype=„multipart/form-data”>
Datoteka: <br />
<input type=„file” name=„datoteka” value=„” /><br />
<input type=„submit” name=„upload_btn” value=„Upload” />
</form>
```

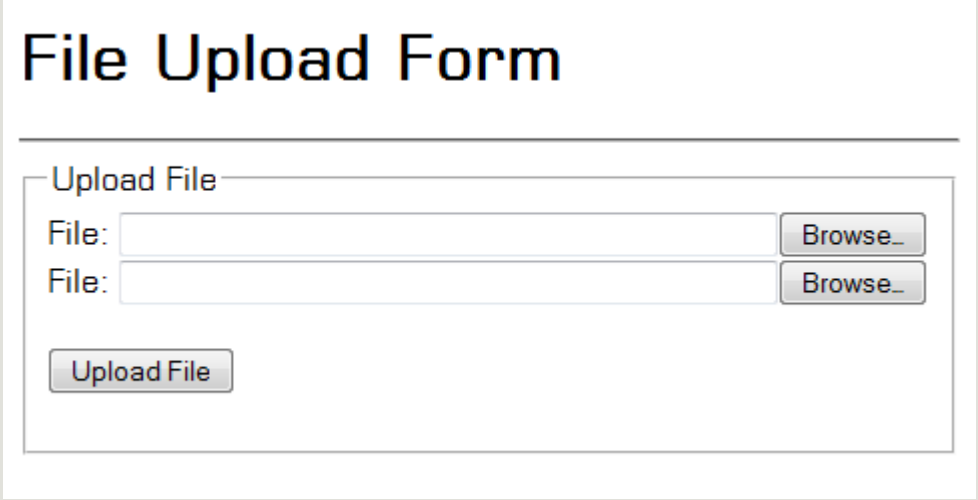


Vrijednost atributa enctype kazuje pregledniku da naša forma sadrži tekstualne podatke i datoteku. Element input za odabir datoteke ima vrijednost atributa type jednaku file pa će preglednik prikazati element input-file kao polje za odabir datoteke sa korisnikovog računala. Na kraju forme nalazi se i gumb Upload za pokretanje prijenosa datoteke na poslužitelj. Gumb Browse... dio je elementa za odabir datoteke s korisnikovog računala i prikazuje ga preglednik.

Polje $_FILES

Podatke o poslanoj datoteci PHP pohranjuje u super-globalnu varijablu $_FILES. Upotrebom funkcije print_r dobije se:

```
Array
(
[datoteka] => Array
    (
        [name] => PHP-test.pdf
        [type] => application/download
        [tmp_name] => /tmp/phpF7D4.tmp
        [error] => 0
        [size] => 98174
    )
)
```



Opis podataka

Ključ datoteka u prvom polju je vrijednost atributa name u polju input za odabir datoteke s korisnikovog računala. Vrijednost tog ključa je novo polje koje sadrži podatak o imenu datoteke, tipu, veličini i lokaciji na koju je web server privremeno pohranio datoteku (najčešće nekakav privremeni folder na poslužitelju koji je definiran u konfiguraciji web-servera).

Da se ne bi obrisala datoteku je nakon prijenosa potrebno prenijeti u neki drugi folder na poslužitelju:

$uploaddir='/doc/'; //definira stalni odredišni direktorij

$uploadfile=basename($_FILES['datoteka']['name']); // izvlači naziv datoteke

$file_array=explode(„.",$uploadfile);//izvlače ekstenziju

$file_ext=end($file_array); //datoteke

$file_onserver=„file_".time()."."."$file_ext;//uključuje vremensku komponentu u naziv datoteke

$new_file_name=$uploaddir.$file_onserver;//stvorena varijabla s novim imenom i novom putanjom

Premještanje datoteke

Uključenjem vremenske komponente rješava se moguć problem ako dva korisnika pošalju dokument s istim nazivom.

if (move_uploaded_file($_FILES[‘datoteka’][‘tmp_name’],$new_file_name)) {//...}

//premještamo datoteku iz privremenog u stalni folder. Funkcija vraća istinu ako je micanje bilo uspješno pa možemo izvršiti neki kod – prikazati obavijest korisniku ili zapisati neki log zapis o obavljenom prijenosu

Rad s datotekama – spajanje na datoteku

Pomoću ugrađenih funkcija PHP može pristupati datotekama na poslužitelju, pročitati sadržaj ili zapisati novi.

Npr, ako je na poslužitelju u mapi gdje se nalazi PHP skripta, tekstualna datoteka gradovi.txt, treba ju otvoriti:

$fp=fopen('gradovi.txt','r');//fopen vraća identifikator konekcije prema datoteci ili laž, parametri su putanja i način rada, odnosno pristupa datoteci

```
if($fp)
{//...}
else
{echo 'GRESKA';}
```

Način	Opis
r	Datoteka se otvara za čitanje. Pokazivač se postavlja na početak.
r+	Datoteci se pristupa za čitanje i pisanje. Pokazivač je na početku.
w	Datoteka se otvara samo za pisanje (ako ne postoji kreira se). Pokazivač je na početku
w+	Pristupa se za čitanje i pisanje (ako ne postoji, kreira se). Pokazivač je na početku
a	Datoteka se otvara samo za pisanje (ako ne postoji kreira se). Pokazivač je na kraju.
a+	Datoteci se pristupa za čitanje i pisanje (ako ne postoji kreira se).Pokazivač je na kraju

Čitanje iz tekstualne datoteke

$sadrzaj='';

while(!feof($fp))

$sadrzaj.=fread($fp, 8192);//parametri su identifikator konekcije i broj bajtova koji će biti pročitan

Ili sav sadržaj odjednom ovisno o veličini datoteke

$sadrzaj=fread($fp, filesize('gradovi.txt');

Ili liniju po liniju

$datoteka=file('gradovi.txt');

foreach($datoteka as $line_num=>$line)

echo 'Linija #'.$line_num.' :'.$line.'
';

Funkcija file ne traži identifikator konekcije i sprema sav sadržaj datoteke u varijablu koja postaje polje podataka pa pomoću foreach petlje prolazimo kroz elemente tog polja, a svaki ključ predstavlja jedan red datoteke.

Zapisivanje u datoteku

```
$fp=fopen('gradovi.txt','w');//ili 'a'
if($fp)
{fwrite($fp, 'Dubrovnik');
$sadrzaj=fread($fp, filesize('gradovi.txt'));
echo $sadrzaj;
}
else
{echo 'GRESKA';}
```

PHP funkcije za rad s mapama i datotekama

Funkcija is_file provjera je li parametar regularna datoteka.

Funkcija is_dir provjerava je li parametar mapa. Korisna je kod prenošenja datoteka na poslužitelj.

Funkcija mkdir omogućava kreiranje mape: mkdir('/var/www/test/'). Vraća istinu ako je operacija kreiranja izvršena uspješno. Drugi parametar može postaviti dozvole (linux).

Funkcija rmdir briše mapu koja postoji. Vraća istinu za uspješno brisanje.

Funkcija unlink briše datoteku čiji su joj naziv i putanja predani preko parametra. Rezultat je istina ako je brisanje bilo uspješno.

Rad s bazama podataka

SQL – programski jezik za komunikaciju s bazom podataka, omogućuje programerima web-aplikacija da koriste različite baze podataka na različitim platformama.

Relacijska baza podataka – relacijski model baze podataka zasnovan je na ideji da se cjelokupni skup podataka razdjeli na pravokutne tablice (relacije) pri čemu jedan stupac relacije obično sadrži vrijednost jednog atributa (za entitet ili vezu) koji čine podaci istog tipa. Svaki redak tablice predstavlja zapis za jednog učenika, a stupci su atributi zapisa.

Stupac id je obično primarni ključ. Primarni ključ omogućuje razlikovanje zapisa, on je vrijednost stupca koja na jedinstven način određuje neki redak.

Neki atributi mogu biti strani ključevi, odnosno primarni ključevi nekih drugih zapisa iz drugih tablica (npr. poštanski broj u tablici mjesto). Pomoću stranih ključeva smanjuje se broj ponavljanja.

SQL osnove

Naredbe za manipulaciju podacima:

SELECT

INSERT

UPDATE

DELETE

S bazom podataka komunicira se preko posebnog sučelja (naredbenog retka ili nekog alata kao što je phpMyAdmin, aplikacije za upravljanje MySQL bazom podataka).

Naredba SELECT

Služi za dohvat podataka iz baze

```
SELECT atr1, atr2, atr3     //nazivi stupaca tablice (atributi) ili * za sve atribute
FROM tablica                //naziv relacije (tablice) iz koje dohvaćamo podatke
WHERE uvjeti                //uvjeti prema kojima izdvajamo podatke
```

Naredba LIMIT

Za lakši prikaz kad SQL upit dohvaća velik broj redova.

SELECT * FROM ucenici LIMIT 0, 2;

Ova naredba uvijek dolazi na kraju i ima dva parametra. Prvi parametar označava od kojeg zapisa, a drugi broj zapisa koji će biti dohvaćeni.

Spajanja – inner join

Spajanja podataka (joins) – jednim upitom dohvaćaju se podaci iz više tablica u jednoj bazi podataka. To omogućuje relacijski model sa primarnim i stranim ključevima.

SELECT ucenici.ime, ucenici.prezime, ucenici.adresa, ucenici.pbr, mjesto.naziv

FROM ucenici, mjesto

WHERE ucenici.pbr=mjesto.pbr; //definirana je veza između dviju tablica

Ovo je tzv. unutarnje spajanje ili inner join.

SELECT ucenici.ime, ucenici.prezime, ucenici.adresa, ucenici.pbr, mjesto.naziv

FROM (ucenici INNER JOIN mjesto ON ucenici.pbr=mjesto.pbr)

Prikazuju se samo zapisi koji imaju poveznicu.

Spajanja – natural (left) join

Prirodno (lijevo) spajanje

SELECT ucenici.ime, ucenici.prezime, ucenici.adresa, ucenici.pbr, mjesto.naziv

FROM (ucenici LEFT JOIN mjesto ON ucenici.pbr=mjesto.pbr)

Dohvaća sve učenike neovisno o tome imaju li definiran atribut pbr.

Subselect

Naredba SELECT unutar naredbe SELECT. Npr. za sve učenike koji su iz Zadra, a ne znamo pbr:

SELECT ime, prezime, adresa

FROM ucenici

WHERE pbr = (SELECT pbr FROM mjesto WHERE naziv LIKE ‘Zadar’)

Oblik sintakse za dohvaćanje adrese nekog učenika:

SELECT ucenici.adresa, ucenici.pbr,

(SELECT mjesto.naziv FROM mjesto WHERE mjesto.pbr=ucenici.pbr) AS mjesto_naziv

FROM ucenici WHERE prezime LIKE ‘Horvat’ AND ime LIKE ‘Ivan’;

Moglo se koristiti i join. Pomoću naredbe AS napravljen je alias ili drugi naziv stupca.

Naredba INSERT

Služi za dodavanje novih zapisa u tablicu u bazi podataka.

INSERT INTO tablica (atr1, atr2, atr3)

VALUES (val1, val2, val3);

Atribut koji je primarni ključ dodaje se automatski u vrijednosti za jedan većoj od najveće.

Za unos više novih zapisa:

INSERT INTO tablica (a1, a2, a3)

VALUES (v1, v2, v3), (v4, v5, v6), (v7, v8, v9);

Naredba UPDATE

Za uređivanje (promjenu) vrijednosti atributa koje su već pohranjene u bazu podataka.

UPDATE tablica

SET atr1=val1, atr2=val2

WHERE uvjet izmjene;

Primjer:

UPDATE ucenici

SET pbr=23000

WHERE id=122;

Naredba DELETE

Za brisanje jednog ili više zapisa iz tablice. Briše sve zapise iz tablice koji zadovoljavaju uvjet brisanja.

DELETE FROM tablica WHERE uvjet brisanja;

Primjeri:

DELETE * FROM ucenici WHERE id=121;

DELETE * FROM ucenici WHERE pbr=10000;

Spajanje na MySQL bazu podataka

Da bi PHP skripta mogla izvoditi SQL naredbe nad MySQL bazom podataka , na bazu se najprije potrebno spojiti.

MySQL baza podataka je računalni program pokrenut u pozadini spreman za konekciju koju može uspostaviti neka vanjska aplikacija ili PHP skripta.

Sintaksa ugrađene PHP funkcije koja omogućuje spajanje je
`mysqli_connect('server','mysql_user','mysql_password');`

Prvi parametar je naziv/adresa poslužitelja ili računala na kojem je instalirana baza podataka, ako je isto na kojem se nalazi PHP skripta onda se piše localhost.

Ostali parametri su korisničko ime i lozinka za pristup bazi i daje nam ih pružatelj web usluga kod kojeg je smještena stranica i skripta. Ako je instanca baze podataka podignuta na vlastitom računalu ove podatke možemo odrediti sami ili ih nije ni potrebno unijeti.

Provjera ishoda spajanja

Ako je konekcija uspješna, funkcija mysql_connect() kao rezultat vraća identifikator MySQL konekcije, inače vraća false.

```
$db=mysqli_connect(‘localhost’, ’admin’, ’fau12rtz’);

if($db)
   echo ‘ Spojeni ste na bazu podataka’;

else

   echo ‘ Doslo je do pogreske prilikom spajanja’;
```

Ako je spajanje s aplikacijom baze podataka bilo uspješno, varijabla $db poprima vrijednost identifikatora MySQL konekcije, inače je false pa se to koristi za provjeru.

Odabir baze podataka

Aplikacija baze podataka može sadržavati više različitih baza, pa je potrebno izvršiti odabir baze pomoću funkcije mysqli_select_db:

mysqli_select_db(‘skola’, $db);

Prvi parametar je naziv baze podataka, drugi sadržava identifikator MySQL konekcije. Ako je veza uspostavljena, funkcija vraća true.

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$database = „skola”;
$conn = mysqli_connect($servername, $username,
$password);
if (!$conn) {
   die(„Neuspjela konekcija: " . mysqli_connect_error());
}
else
{
echo „Uspjesna konekcija.";
$db_selected=mysqli_select_db($database, $conn);

if($db_selected)
  echo ‘ Spojeni ste na odabranu bazu podataka’;
else
  echo ‘ Doslo je do pogreske prilikom spajanja’;
}
?>
```

Upotreba naredbe include

Funkcija za uključivanje skripte unutar druge skripte.

Kod jedne skripte napiše se u posebnu datoteku (npr. db_connection.php) i onda poziva u drugoj skripti:

```
<?php

include („db_connection.php”);

?>
```

Skripta koju uključujemo treba biti u istoj mapi kao i skripta koja radi s podacima iz baze.

Oblikovanje i izvršavanje SQL upita

Nakon ostvarivanja veze s bazom, formira se SQL upit prema bazi. Za izvršavanje SQL upita koristi se ugrađena funkcija msqli_query.

mysqli_query(„SELECT * FROM ucenici”, $conn);

Ova funkcija ima dva parametra: SQL upit i identifikator MySQL konekcije.

Pri radu s jednom bazom podataka drugi je parametar opcionalan.

Ako je došlo do pogreške pri izvršenju upita, funkcija vraća false. Kod upita INSERT, UPDATE i DELETE, funkcija vraća true ako nije bilo pogreške, inače uglavnom vraća dohvaćene podatke.

SQL upit može se spremiti u posebnu varijablu: $query=„SELECT * FROM ucenici”;

Rezultat izvršenja može se provjeriti načinom čestim u praksi preko
$result=mysqli_query($query);

Dohvaćanje broja vraćenih redaka

Prilikom upotrebe naredbe SELECT, pomoću funkcije mysqli_num_rows može se saznati koliko je zapisa (redaka) za naš upit baza (‘učenici’) vratila:

echo ‘ Broj redova: ‘.mysqli_num_rows($result);

Primjer izvršavanja naredbe INSERT

Oblikovanje duljeg SQL upita:

$query = „INSERT INTO ucenici”;

$query .= „(ime, prezime, adresa, pbr, datuRod, OIB)”;

$query .= „VALUES”;

$query .= „(‘Mario’, ‘Novak’, ‘Put 1’, ‘23000’ ”;

$query .=„’1984-10-12’, ‘1456895756’)”;

$result = myslqli_query($query);

Dohvaćanje broja promijenjenih redaka

Funkcija mysqli_num_rows ne djeluje ako $result vraća true pa se koristi funkcija mysqli_affected_rows (bez parametara) koja vraća broj promijenjenih redova uzimajući podatke i trenutno aktivne konekcije na bazu podataka:

```
$affected_rows = mysqli_affected_rows();
```

Dohvaćanje podataka funkcijom mysqli_fetch_row

Funkcija koja vraćene podatke pretvara u indeksno polje:

```
include („db_connection.php”);

$query = „SELECT ime, prezime ”;

$query .= „FROM ucenici”;

$query .=„WHERE id>120”;

$result = myslqli_query($query);

while($row=msqli_fetch_row($result))
        echo ‘ime: ‘.$row[0].’ prezime: ‘.$row[1].’<br />’;
```

Izraz u zagradi istinit je sve dok varijabla $row može poprimiti vrijednost nekog retka tablice iz baze. Sa svakim prolazom petlje dohvaća se novi red i ispisuju vrijednosti podataka iz dohvaćenog retka (ime i prezime, row predstavlja ključ).

Dohvaćanje podataka funkcijom mysqli_fetch_array

Kod velike količine podataka prihvatljivije je koristiti mysqli_fetch_array:

```
while($row=msqli_fetch_array($result))
        echo ‘ime: ‘.$row[‘ime’].’ prezime: ‘.$row[‘prezime’].’<br />’;
```

Ova funkcija dohvaća red po red i dohvaćene vrijednosti stavlja u polje čiji su ključevi i asocijativni i indeksni pa se podacima može pristupiti preko naziva atributa ili preko indeksa polja.

Funkcija mysqli_fetch_assoc

Dohvaćene podatke stavlja u asocijativno polje:

```
while($row=msqli_fetch_assoc($result))
        echo ‘ime: ‘.$row[‘ime’].’ prezime: ‘.$row[‘prezime’].’<br />’;
```

Ne omogućuje korištenje indeksa za pristup podacima jer stvara isključivo asocijativno polje.

Ako pritom u SELECT upitu mijenjamo nazive atributa pomoću ključne riječi AS, vrijednosti atributa pristupamo preko novog naziva:

```
$query = „SELECT ime AS i, prezime AS p”;

$query .= „FROM ucenici”;

$query .=„WHERE id>120”;

$result = myslqli_query($query);

while($row=msqli_fetch_assoc($result))
        echo ‘ime: ‘.$row[i].’ prezime: ‘.$row[p].’<br />’;
```

Funkcija msqli_fetch_object

Funkcija koja rezultat dohvaćanja stavlja u objekt:

```
while($row=msqli_fetch_object($result))
        echo ‘ime: ‘.$row->ime.’ prezime: ‘.$row->prezime’<br />’;
```

Prikaz dohvaćenih podataka na stranici

Kombiniranje dohvaćanja podataka iz baze i prikaz sa HTML kodom - stvaranje padajućeg izbornika s nazivima mjesta iz tablice mjesto (pbr, naziv):

```
include („db_connection.php”);

$query = „SELECT pbr, naziv FROM mjesto”;

$query .=„ORDER BY naziv ASC”;

$result = mysqli_query($query);

echo ‘<select name=„mjesto”>’;

while($row=mysqli_fetch_array($result)) //ili

while($row=my_fetch_object($result))

{

echo ‘<option value=„’.$row[„pbr”].’”>’;

echo $row[„naziv”];

echo ‘</option>’;

}

echo ‘</select>’;
```

Konekcija na više baza podataka odjednom

Kad se jedna PHP skripta treba spojiti na više baza odjednom (radi sinkronizacije ili prebacivanja podataka) potrebno je kreirati dvije skripte, za svaku konekciju po jednu.

Upravljanje pogreškama

Pogreška može stvoriti sigurnosnu rupu za našu stranicu ili aplikaciju. Korisnici i posjetitelji trebaju samo znati da je do pogreške došlo, bez suvišnih tehničkih detalja.

Upotreba i oblikovanje ispisa funkcijom die

Zaustavlja izvođenje skripte kada dođe do pogreške prikazujući proizvoljni tekst:

$result = mysql_query($query) or die (‘Doslo je do pogreske!’);

Funkcija može ispisati bilo koji format koda.

$result1 = mysql_query($query) or die (‘Pogreska!’);

Upotreba funkcija mysqli_errno i mysqli_error

Pomažu prikazati više informacija o pogrešci koja se dogodila.

mysqli_errno() vraća broj pogreške koja se dogodila u SQL upitu

mysqli_error() vraća tekstualni opis pogreške koja se dogodila u SQL upitu

```
include („db_connection.php”);

$query = „SELECT imeUcenik, prezime FROM ucenici”;

$query .=„WHERE pbr=10000”;

$result = mysqli_query($query);

if ($result)

{

...

}

else

{

echo „Doslo je do pogreske: „;

echo mysqli_errno().”: „.mysqli_error().”\n”;

}
```

Objektno orijentiran pristup bazi podataka

Može se koristiti ugrađena klasa MySQLi (improved, brža komunikacija s bazom, brži rad s podacima iz baze podataka)

Klasa omogućuje proceduralni način rada ili OOP, bržu obradu podataka radi podržanih novih protokola u komunikaciji sa bazom, pripremljeni upiti smanjuju količinu podataka koja se prenosi između skripte i SQL servera, omogućuje i napredne mogućnosti u radu s konekcijama prema bazi podataka, do 40 puta brže izvršavanje upita u nekim slučajevima, nudi unaprijeđenu sigurnost s mogućnošću prepoznavanja i odbacivanja potencijalno opasnih upita, te bolje upravljanje pohranom zaporki u bazi.

Proceduralni pristup - primjer

```
<?php
$link=mysqli_connect(
'localhost', '', 'p8ss1234', 'skola');

if (!$link)
 {
   echo 'Pogreska u konekciji:';
   echo mysqli_connect_error();
   exit();
}
 $query=„SELECT * FROM mjesto LIMIT 5";
 if($result=mysqli_query($link, $query))
{
while ($row=mysqli_fetch_assoc($result))
  {
      echo $row[„nazMjesto"];
      echo '<br />';
   }
 }
 mysqli_close($link);
 ?>
```

OO pristup

Prikazuje prave mogućnosti nove poboljšane klase. Prvo je potrebno instancirati objekt iz klase pozivom konstruktora sa parametrima pomoću kojih se inicijalizira konekcija prema bazi podataka:

```
<?php
$mysqli = new mysqli (
'localhost', '', 'p8ss1234', 'skola');

$query=„SELECT * FROM mjesto LIMIT 5”;
if($result=$mysqli->query($query))
//pozvana je metoda query sa upitom kao parametrom, rezultat je novi objekt $result
{
while ($row=$result->fetch_assoc())
//vraća polje podataka koje predstavlja dohvaćeni zapis iz tablice
 {
      echo $row[„nazMjesto”];
      echo '<br />';
  }
}
mysqli_close();
//zatvara konekciju prema bazi
?>
```

Pripremljeni upiti

Mogućnost pisanja boljih, bržih i sigurnijih upita koji rezultiraju podizanjem performansi aplikacije ili stranice. Mogući su pripremljeni upiti s vezanim parametrima i pripremljeni upiti s vezanim rezultatima.

Vezani parametri u pripremljenim upitima

Mogućnost stvaranja predloška tijela SQL upita koji se nakon provjere sintaktičke ispravnosti pohranjuje na MySQL server, tako da se prilikom izvršenja upita serveru prosljeđuju samo podaci (parametri) koji moraju popuniti predložak.

Kod izvršavanja upita upotrebljava se njegov identifikator. Server formira cjelokupan SQL upit i izvršava ga.

Jedan se predložak može upotrebljavati više puta, što bitno smanjuje količinu poslanih podataka čime se komunikacija s MySQL serverom ubrzava.

Primjer - predložak upita za dodavanje novog mjesta u tablicu:

INSERT INTO mjesto (pbr, naziv) VALUES (?,?);

Na mjestu podataka koji se pohranjuju u tablicu stavlja se znak ? kako bi rezervirao mjesto u predlošku za podatke koji će doći.

Primjer upotrebe predloška SQL upita

```
<?php
$mysqli = new mysqli(
'localhost', '', 'p8ss1234', 'skola');

if (mysqli_connect_errno())
 {
   echo 'Pogreska u konekciji:';
   echo mysqli_connect_error();
   exit();
}

 $query_tp1="SELECT naziv FROM mjesto WHERE pbr=?";
$pbr="31000"

if($stmt=$mysqli->prepare($query_tp1))
//stvara novi objekt
{
$stmt->bind_param('i',$pbr);
//šalje serveru podatak kojim se popunjava predložak
$stmt->execute();
//izvršava se SQL upit na serveru
$stmt->bind_result($naziv);
//gdje će se pohraniti rezultat upita
$stmt->fetch();
//dohvaća podatke i sprema ih
echo $naziv;
$stmt->close();
 }

 mysqli->close();
 ?>
```

Primjer upotrebe SQL predloška – naredba INSERT

```
<?php
$mysqli = new mysqli(
'localhost', 'admin', 'p8ss1234', 'skola');

if (mysqli_connect_errno())
 {
   echo 'Pogreska u konekciji:';
   echo mysqli_connect_error();
   exit();
}

$query_tpl = „INSERT INTO mjesto (pbr, naziv) „;
$query_tpl .= VALUES (?,?)";
 $pbr=„21000";
 $naziv=„Split";

if($stmt=$mysqli->prepare($query_tpl))
{
$stmt->bind_param('is', $pbr, $naziv);
//ugrađuju se podaci na predviđeno mjesto
$stmt->execute();
$stmt->close();
 }

 mysqli->close();
 ?>
```

Metoda bind_param()

Prvi parametar služi za specifikaciju tipova podataka za parametre koji slijede.

‘i’ znači da je parametar cjelobrojnog tipa

‘is’ znači da je prvi parametar cijeli broj, a drugi niz znakova ili string

Za svaki parametar u prvom parametru metode mora postojati odgovarajuće slovo koje određuje njegov tip čime se povećava sigurnost jer se MySQLi klasa brine da tip odgovara definiranom.

Svaki je tip predstavljen slovom (d je za double ili float).

Vezani rezultati u pripremljenim upitima

Određeni atribut iz tablice veže se za određenu varijablu.

```
<?php
$mysqli = new mysqli(
'localhost', 'admin', 'p8ss1234', 'skola');

if (mysqli_connect_errno())
 {
   echo 'Pogreska u konekciji:';
   echo mysqli_connect_error();
   exit();
}

$query = „SELECT pbr, naziv FROM mjesto „;
$query .= „WHERE naziv LIKE 'Z%'";

 if($stmt=$mysqli->prepare($query))
{
$stmt->execute();
$stmt->bind_result($col1, $col2);

while($stmt->fetch())
//prolazi kroz sve dohvaćene retke tablice
{
                echo $col1." – „.$col2;
                echo '<br />';
}

$stmt->close();

  }

  mysqli->close();
  ?>
```

Kombinacija vezanih parametara i rezultata

```
<?php
$mysqli = new mysqli(
'localhost', 'admin', 'p8ss1234', 'skola');

if (mysqli_connect_errno())
 {
   echo 'Pogreska u konekciji:';
   echo mysqli_connect_error();
   exit();
}

$query = „SELECT pbr, naziv FROM mjesto „;
$query .= „WHERE naziv LIKE '?%'";
$slovo = 'Z';

 if($stmt=$mysqli->prepare($query))
{
$stmt->bind_param('s', $slovo);
//slovo ugrađuje unutar predloška
$stmt->execute();
$stmt->bind_result($col1, $col2);

while($stmt->fetch())
{
                echo $col1." – „.$col2;
                echo '<br />';
}

$stmt->close();

 }

 mysqli->close();
 ?>
```