

Współpraca mysqli z bazą (wariant obiektowy)¹

Nawiązanie połączenia

Wariant proceduralny (z użyciem funkcji):

```
require_once "dbconnect.php";  
$conn = mysqli_connect($host, $user, $pass, $db);
```

Wariant obiektowy:

```
require_once "dbconnect.php";  
$conn = new mysqli($host, $user, $pass, $db);
```

Nasze połączenie `$conn` to nowy obiekt, tworzony z użyciem operatora `new` (ang. nowy). Obiekt będzie posiadał tzw. metody (czyli funkcje przewidziane w przepisie klasy, wywoływane na rzecz naszego obiektu) oraz atrybuty (właściwości, zmienne charakteryzujące obiekt). Natomiast zapis prezentowany powyżej nazywamy konstruktorem (gdyż rzeczywiście następuje tu konstrukcja, czyli stworzenie obiektu).

Obsługa polskich znaków

Wariant proceduralny:

```
mysqli_set_charset($conn, "utf8");
```

Wariant obiektowy:

```
$conn->set_charset("utf8");
```

W wersji obiektowej ustawienie zestawu znaków `utf8` realizowane jest z użyciem metody `set_charset()` wywołanej na rzecz naszego obiektu `$conn`. Koniecznie zwróćmy uwagę na operator `->`, który jest jak to mówimy dwuargumentowy, to znaczy posiada coś po lewej swojej stronie, jak i po prawej. I w przypadku metod obiektowych `mysqli`, po lewej stronie operatora `->` znajdzie się nasz obiekt `$conn` reprezentujący połączenie z bazą, zaś po prawej stronie znajdzie się odpowiednia metoda (tutaj `set_charset()`, ustawiająca zestaw znaków `utf8`). Metoda tym różni się od zwykłej funkcji, iż jej definicja jest częścią klasy, która opisuje obiekt, przez co może zawierać lepsze mechanizmy hermetyzacji (chronienia, ukrywania) danych oraz obsługi błędów.

Wykonanie zapytania SQL

Wariant proceduralny:

```
$q = "SELECT id, nazwa FROM klasa";  
$result = mysqli_query($conn, $q);
```

¹ <https://pasja-informatyki.pl/programowanie-webowe/php-mysqli-wariant-obiektowy/>

Wariant obiektowy:

```
$q = "SELECT id, nazwa FROM klasa";  
$result = $conn->query($q);
```

W wersji obiektowej nie potrzebujemy dwóch argumentów (identyfikatora połączenia oraz treści zapytania), gdyż metoda `query()` wywołana jest przecież na rzecz naszego obiektu `$conn` (stąd wystarczy jedynie treść zapytania, umieszczona w zmiennej `$q`).

Pobranie ilości zwróconych rekordów

Wariant proceduralny:

```
$result = $conn-> query($q);  
$ile = mysqli_num_rows($result);
```

Wariant obiektowy:

```
$result = $conn-> query($q);  
$ile = $result->num_rows;
```

W tym przypadku mamy do czynienia nie z metodą obiektu, tylko z jego atrybutem `num_rows` (zwróćmy uwagę na brak nawiasów okrągłych właściwych dla funkcji). Liczba rekordów zostaje umieszczona w atrybucie `num_rows` (czyli zmiennej przypisanej do obiektu `$conn` w klasie).

Fetchowanie pojedynczych rekordów

W artykule dotyczącym wariantu proceduralnego biblioteki `mysqli`, poznaliśmy trzy sposoby fetchowania rekordów do tablicy jednowymiarowej:

- `mysqli_fetch_row()` - wyjmowanie liczbowe - ponumeruj (od zera) szufladki tablicy jednowymiarowej;
- `mysqli_fetch_assoc()` - wyjmowanie asocjacyjne (czyli skojarzeniowe) - ponazywaj słownie szufladki tablicy jednowymiarowej, nadając im zawsze taką samą nazwę, jaką noszą kolumny w bazie danych;
- `mysqli_fetch_array()` - zarówno ponumeruj jak i ponazywaj szufladki tablicy jednowymiarowej (wada: redundancja danych - każda szufladka jest "podwójna", gdyż istnieje zarówno komórka zaindeksowana numerycznie jak i słownie).

Przykład użycia `mysqli_fetch_row()`:

```
$q = "SELECT id, nazwa FROM klasa";  
$result = mysqli_query($conn, $q);  
while($row = mysqli_fetch_row($result))  
{  
    echo $row[0]."<br>"; // $row[0] wypisze id klasy  
}
```

W zapisie obiektowym możemy pozbyć się (dla każdej z tych trzech metod) przedrostka `mysqli_`, a ponadto zamiast zapisania `$result` jako argumentu funkcji, umieszczamy go po lewej stronie operatora `->` (jako obiekt, na rzecz którego wywołujemy metodę fetchującą pojedynczy rekord):

```
$q = "SELECT id, nazwa FROM klasa";
$result = $conn-> query($q);
while($row = $result->fetch_row())
{
    echo $row[0]."<br>"; // $row[0] wypisze id klasy
}
```

Dodatkowo, poznajmy także czwarty sposób dostępu do pojedynczych rekordów. Będzie to metoda `fetch_object()`, która umieszcza wyjęty rekord w obiekcie `$obj` posiadającym atrybuty o takich samych nazwach jak kolumny w bazie:

```
$q = "SELECT id, nazwa FROM klasa";
$result = $conn-> query($q);
while($obj = $result->fetch_object())
{
    echo $obj->id."<br>";
    echo $obj->nazwa."<br>";
}
```

Zamknięcie połączenia z bazą

Brak linii zamykającej połączenie z bazą danych może spowodować utratę punktów na egzaminie - nie zapomnijmy dokonać tego w odpowiednim miejscu kodu, po wszystkich wykonanych bazodanowych czynnościach!

Wariant proceduralny:

```
mysqli_close($conn);
```

Wariant obiektowy:

```
$conn->close();
```

Uwierzytelnienie za pomocą PHP i MySQL²

Niezależnie od tego, gdzie będziemy przechowywać hasła, to obowiązkowo należy je zaszyfrować. Podczas zapisywania haseł w pliku oraz ich odczytu, stosowaliśmy szyfrowanie jednostronne, za pomocą funkcji `md5()`. PHP dysponuje również silniejszą funkcją `sha1`. W

² http://www.zs2krasnystaw.edu.pl/dla_ucznia/kurs_php/k_12_0_0.php#cwicz_12_0_0_1

przypadku zapisywania hasła w bazie danych możemy również zastosować funkcję MySQL o tej samej nazwie SHA1().

Utworzenie przykładowej bazy z hasłami użytkowników

Należy zastosować następujący kod SQL:

```
CREATE DATABASE uzytkownicy;
USE uzytkownicy;
CREATE TABLE hasla
(
    uzytkownik VARCHAR(20) NOT NULL,
    haslo VARCHAR(40) NOT NULL,
    PRIMARY KEY(uzytkownik)
);
INSERT INTO hasla VALUES
('uzytkownik1',SHA1('haslo1')),
('uzytkownik2',SHA1('haslo2'));
```

Logowanie na stronie za pomocą MySQL - metoda proceduralna

Po wpisaniu do formularza loginu i hasła i wciśnięciu Zaloguj, powinniśmy otrzymać odpowiedni komunikat.

Login Hasło

Jesteś zalogowany

Oto plik logowanie.php:

```
<!doctype html>
<html>
    <head>
        <title>Logowanie z wykorzystaniem MySQL</title>
        <meta charset="utf-8" />
    </head>
    <body>
        <form action="logowanie.php" method="post">
            Login<input type="text" name="login" />
            Hasło<input type="password" name="haslo" />
            <input type="submit" name="logowanie" value="Zaloguj" />
        </form>
        <br />

    <?php
```

```

//jeżeli nie ma wysłanych danych, to zakończ skrypt
if (!isset($_POST['login']) || !isset($_POST['haslo']))
exit;
$login=trim($_POST['login']);
$haslo=trim($_POST['haslo']);
if (empty($login) || empty($haslo))
{
    echo 'Brak loginu lub hasła!';
    exit;
}
$mysql=mysqli_connect('localhost','admin','admin123');
//połączenie z MySQL
if (!$mysql)
{
    echo 'Brak połączenia z MySQL!';
    exit;
}
$baza=mysqli_select_db($mysql,'uzytkownicy');
//wybór potrzebnej bazy
if (!$baza)
{
    echo 'Brak połączenia z bazą użytkowników!';
    exit;
}
$haslo=sha1($haslo);//zaszyfrowanie hasła
//znajdowanie pasującego wiersza
$zapytanie="select count(*) from hasla where
uzytkownik='$login' and haslo='$haslo'";
$wynik=mysqli_query($mysql,$zapytanie);
if (!$wynik)
{
    echo 'Błąd w wykonaniu zapytania!';
    exit;
}
$wiersz=mysqli_fetch_row($wynik);
$ile_znaleziono=$wiersz[0];
if ($ile_znaleziono>0) echo 'Jesteś zalogowany';
else echo 'Podałś błędny login lub hasło!';
?>
</html>

```

- Zwróć uwagę, że w ujęciu proceduralnym operujemy nie na obiektach, lecz na zasobach. Uchwyty do tych zasobów przekazujemy jako parametry funkcji o nazwach rozpoczynających się od `mysqli...`. Takim zasobem jest u nas `$mysql`.

[Logowanie na stronie za pomocą MySQL - podejście obiektowe](#)

Wykonamy ponownie poprzednie ćwiczenie, stosując tym razem obiektowy interfejs biblioteki `mysqli`.

Plik logowanie.php, przyjmie tym razem postać:

```
<!doctype html>
<html>
  <head>
    <title>Logowanie z wykorzystaniem MySQL</title>
    <meta charset="utf-8" />
  </head>
  <body>
    <form action="logowanie.php" method="post">
      Login<input type="text" name="login" />
      Hasło<input type="password" name="haslo" />
      <input type="submit" name="logowanie" value="Zaloguj"
/>
    </form>
    <br />

    <?php
      //jeżeli nie ma wysłanych danych, to zakończ skrypt
      if (!isset($_POST['login']) || !isset($_POST['haslo']))
exit;
      $login=trim($_POST['login']);
      $haslo=trim($_POST['haslo']);
      if (empty($login) || empty($haslo))
      {
        echo 'Brak loginu lub hasła!';
        exit;
      }
      //połączenie z MySQL
      @ $baza=new mysqli('localhost','root','','uzytkownicy');
      if (mysqli_connect_errno())
      {
        echo 'Brak połączenia z MySQL!';
        exit;
      }
      $haslo=sha1($haslo);//zaszyfrowanie hasła
      //znajdowanie pasującego wiersza
      $zapytanie="select * from hasla where uzytkownik='$login'
and haslo='$haslo'";
      $wynik=$baza->query($zapytanie);
      $ile_znaleziono=$wynik->num_rows;
      if ($ile_znaleziono>0) echo 'Jesteś zalogowany';
      else echo 'Podałeś błędny login lub hasło!';
      $wynik->free();
      $baza->close();
    ?>
  </html>
```