

Podejście obiektowe.

1. Korzystanie z PDO

PDO z ang. **PHP Data Objects**, to rozszerzenie pozwalające na ujednoliconą obsługę różnych baz danych w technice obiektowej. Jest częścią **PHP** począwszy od wersji 5.1. Jego zaletą jest ujednolicona obsługa połączeń z bazami, niezależnie od tego, czy będzie to **SQLite**, **MySQL**, **PostgreSQL**, czy dowolna inna.

1.1 Nawiązanie połączenia

Nawiązanie połączenia z bazą wygląda tu zupełnie inaczej niż w klasycznym podejściu. Należy w tym celu utworzyć nowy obiekt klasy **PDO**, korzystając z konstruktora w postaci:

`PDO(źródło_danych[, użytkownik[, hasło[, opcje]]]);`

Gdzie **źródło_danych** to ciąg znaków określający rodzaj bazy danych i sposób połączenia, użytkownik to nazwa użytkownika, hasło to hasło, a opcje to tablica zawierająca dodatkowe opcje związane z połączeniem. Jedynym wymaganym argumentem jest **źródło_danych**, pozostałe są opcjonalne.

Uproszczona postać ciągu **źródło_danych** dla bazy **MySQL**, z której będziemy korzystać, ma postać:

`mysql:host=nazwa_serwera;port=numer_portu;dbname=nazwa_bazy`

Poszczególne składowe mają następujące znaczenie:

- **nazwa_serwera** – określenie nazwy lub adresu serwera baz danych (np. localhost)
- **numer_portu** – port na którym ma nastąpić połączenie z bazą. Może być pominięty, zostanie wtedy zastosowana wartość standardowa.
- **nazwa_bazy** – nazwa bazy danych, z którą ma nastąpić połączenie.

Tak więc ciąg **źródło_danych**, który pozwala na połączenie z bazą **testphp** znajdującą się na serwerze **MySQL** pracującym na komputerze lokalnym na porcie **3306**, będzie miał postać:

`mysql:host=localhost;port=3306;dbname=testphp`

lub, jeśli określenie portu zostanie pominięte

`mysql:host=localhost;dbname=testphp`

Jeżeli wywołanie konstruktora zakończy się sukcesem, zostanie zwrócony obiekt, który będzie służył do dalszej komunikacji, należy więc przypisać go jakiejś zmiennej. W przypadku gdy połączenia nie uda się nawiązać, zostanie wygenerowany wyjątek typu **PDOException**, który będzie zawierał opis przyczyny powstania błędu. A zatem fragment kodu ustalający, czy udało się nawiązać połączenie z bazą mógłby mieć postać:

```

<?php
$dsn = "mysql:host=localhost;dbname=testphp";
$uzytkownik = "php";
$haslo = "test";
try{
    $dbo = new PDO($dsn, $uzytkownik, $haslo);
}
catch (PDOException $e){
    echo 'Błąd połączenia: ' . $e->getMessage();
    exit;
}
//dalsze instrukcje skryptu
?>

```

1.2 Kończenie połączenia

Połączenie nawiązane za pomocą obiektu **PDO** pozostaje aktywne przez cały czas życia obiektu i jest kończone przy usuwaniu obiektu z pamięci. Zostanie to wykonane automatycznie po zakończeniu pracy skryptu lub po przypisaniu wartości **null** zmiennej obiektowej przechowującej odwołanie do obiektu.

1.3 Zapytania pobierające dane

Zapytania są wykonywane za pomocą metody **query**, której należy przekazać treść zapytania w postaci argumentu. Zakładając zatem, że mamy do dyspozycji obiekt **\$dbo** powstały przez wywołanie konstruktora klasy **PDO**, zapytanie **SQL** można wykonać przez zastosowanie konstrukcji:

```
$dbo->query(„treść zapytania”);
```

Oczywiści treść zapytania może być również przekazywana w postaci zmiennej, np.:

```
$query= „treść zapytania”;
```

```
$dbo->query($query);
```

Metoda **query** zwraca obiekt typu **PDOStatement** pozwalający na odczyt danych po wykonaniu zapytania, gdy zakończyło się sukcesem, lub też wartość **false** w przeciwnym razie.

Wspomniany obiekt zawiera metodę **fetch**, która udostępnia pobrane dane. Jej ogólne wywołanie ma postać:

```
fetch([typ_wyniku])
```

Zwracaną wartością jest kolejny wiersz z wyników zapytania lub wartość **false**, jeżeli kolejnego wiersza nie uda się pobrać (np. zostały już odczytane wszystkie dane). Postać zwróconych danych zależy od stanu argumentu **typ_wyniku**, który może przyjmować następujące wartości:

- ***PDO::FETCH_ASSOC*** – zwraca tablicę asocjacyjną, w której nazwy kolumn wynikowych są kluczami,
- ***PDO::FETCH_BOTH*** – zwraca tablicę indeksowaną zarówno numerycznie, jak i asocjacyjnie (jest to wartość domyślna),
- ***PDO::FETCH_BOUND*** – zwraca wartość ***true*** oraz przypisuje wartość z kolumn wyniku do zmiennych ***PHP*** ustalonych wcześniej za pomocą wywołania metody ***bindColumn()***,
- ***PDO::FETCH_CLASS*** – zwraca nową instancję klasy, dokonując mapowania kolumn wynikowych na właściwości klasy,
- ***PDO::FETCH_INTO*** – uaktualnia istniejącą instancję klasy, dokonując mapowania kolumn wynikowych na właściwości klasy,
- ***PDO::FETCH_LAZY*** – kombinacja ***PDO::FETCH_BOTH*** i ***PDO::FETCH_OBJ***,
- ***PDO::FETCH_NUM*** – zwraca tablicę indeksowaną numerycznie,
- ***PDO::FETCH_OBJ*** – zwraca obiekt z właściwościami o nazwach i wartościach odpowiadającym kolumnom wynikowym zapytania.

Aby zmienić domyślny tryb obowiązujący dla wszystkich zapytań (czyli standardowe ***PDO::FETCH_BOTH***), należy wykorzystać metodę ***setFetchMode***, której wywołanie ma postać:

```
setFetchMode([domyślny_typ_wyniku])
```

gdzie ***domyślny_typ_wyniku*** to jedna z wartości wymienionych wyżej. Jeśli chcemy na przykład, aby domyślnym typem wyniku była tablica indeksowana numerycznie, zastosujemy wywołanie:

```
$result->setFetchMode(PDO::FETCH_NUM);
```

Po którym wszystkie wywołania metody ***fetch*** będą zwracały właśnie tablice numeryczne.

Ponieważ każde wywołanie ***fetch*** powoduje zwrócenie kolejnego wiersza wyniku lub wartości ***false***, jeśli zostały odczytane wszystkie wiersze metodę tę można wywoływać w pętli ***while*** o schematycznej postaci:

```
while($row=$result->fetch()){
```

```
//instrukcje przetwarzające wiersz tabeli
```

```
}
```

Czas zatem zebrać przedstawione dotychczas w tej lekcji informacje i zapisać skrypt odczytujący za pomocą ***PDO*** dane z wybranej tabeli bazy danych.

```

<?php
$dsn = "mysql:host=localhost;dbname=testphp";
$uzytkownik = "php";
$haslo = "test";

try{
    $dbo = new PDO($dsn, $uzytkownik, $haslo);
}

catch (PDOException $e){
    echo 'Błąd połączenia: ' . $e->getMessage();
    echo '</div></body></html>';
    exit;
}

$query = "SELECT * FROM osoba";
$result = $dbo->query($query);

if(!$result){
    echo "Nie mogę wykonać zapytania.<br />";
    echo '</div></body></html>';
    exit;
}

?>
<table>
<tr>
    <td>Id</td><td>Imie</td><td>Nazwisko</td>
    <td>Rok urodzenia</td><td>Miejsce urodzenia</td>
</tr>

<?php
while($row = $result->fetch(PDO::FETCH_NUM)){
    echo "<tr>\n";
    echo "<td>{$row[0]}</td>\n";
    echo "<td>{$row[1]}</td>\n";
    echo "<td>{$row[2]}</td>\n";
    echo "<td>{$row[3]}</td>\n";
    echo "<td>{$row[4]}</td>\n";
    echo "</tr>\n";
}

$dbo = null;

?>

```

Warto w tym miejscu pokazać, jak wyglądałby taki skrypt, gdyby wyniki zapytania były by pobierane jako obiekty – taka technika jest często spotykana. W takiej sytuacji wartością zwracaną przez metodę *fetch* jest obiekt, którego właściwości odpowiadają kolumnom

wynikowym zapytania *SQL*. To znaczy nazwami właściwości są nazwy kolumn, a ich wartościami – wartości zapisane w tych kolumnach. A zatem pętla *while* generująca kolejne komórki tabeli *HTML* o zawartości pobieranej z wyników zapytania miałyby w tej technice następującą postać:

```
<?php
while($row = $result->fetch(PDO::FETCH_OBJ)) {
    echo "<tr>\n";
    echo "<td>{$row->Id}</td>\n";
    echo "<td>{$row->Imie}</td>\n";
    echo "<td>{$row->Nazwisko}</td>\n";
    echo "<td>{$row->Rok_urodzenia}</td>\n";
    echo "<td>{$row->Miejsce_urodzenia}</td>\n";
    echo "</tr>\n";
}
$db = null;
?>
```

1.4 Zapytania modyfikujące dane

Do wykonywania zapytań modyfikujących używa się metody *execute*. Wysyła ona zapytanie do serwera oraz zwraca wartość całkowitą określającą liczbę rekordów, na które to zapytanie miało wpływ (czyli np. liczbę zmodyfikowanych, usuniętych lub dodanych wierszy tabeli). Jeżeli wykonanie zapytania nie powiodło się, metoda zwraca wartość *false*. Zapytanie należy przekazać w postaci argumentu, zatem schematyczne wywołanie wygląda następująco:

```
$db->execute(„Treść zapytania”);
```

Zobaczmy więc, jak w praktyce dodać wiersz do tabeli osoba.

```
<?php
$dsn = "mysql:host=localhost;dbname=testphp";
$uzytkownik = "php";
$haslo = "test";
try{
    $dbo = new PDO($dsn, $uzytkownik, $haslo);
}
catch (PDOException $e){
    echo 'Błąd połączenia: ' . $e->getMessage();
    exit;
}

$query = "INSERT INTO osoba VALUES(";
$query .= "0, 'Maciej', 'Nowak', 1982, 'Lublin'";
$query .= ")";

$result = $dbo->exec($query);

if($result === false){
    echo 'Nie mogę wykonać zapytania.';
}
else{
    echo 'Liczba dodanych rekordów: ' . $result;
}
$dbo = null;
?>
```

Zadanie

Napisz skrypt, który będzie zapamiętywał dane dotyczące odwiedzających witrynę użytkowników: datę i czas odwiedzin, adres IP, rodzaj przeglądarki. Informacje te mają być następnie wyświetlane na stronie.