Laborator 02 - BD in PHP

Clase in PHP

Clasele in PHP (versiunea 5) sunt foarte asemanatoare cu cele din JAVA. PHP suporta mostenire (ca si in JAVA, in PHP nu ofera suport pentru mostenire multipla, altfel spus o clasa copil nu poate mosteni decat o singura clasa parinte). De asemenea, in PHP se pot defini clase abstracte, interfete, metode statice. PHP suporta polimorfismul si supraincarcarea. Pentru mai multe detalii legate de Clase si Obiecte in PHP 5, consultati: http://us2.php.net/oop
Exemplu de clasa in PHP 5:

```
class Example {
   private $id;
   public $string;
   public function display () {
      echo $string;
   }
}
```

Constructorii se pot declara folosind doua modalitati in PHP 5:

```
class Example {
   private $id;
   public $string;
   public function Example ($id, $string) {
   //TODO: constructor code
   }
}
```

Aceasta varianta este perfect asemanatoare cu declararea unui constructor in Java. Pe langa aceasta mai exista si varianta:

```
class Example {
   private $id;
   public $string;
   public function __construct($id, $string) {
   //TODO: constructor code
   }
}
```

Aceasta varianta foloseste cuvantul-cheie **construct**. Avantajele acestei metode sunt legate de numele clasei. **construct** pune in evidenta aspectul comportamental al metodei (si face pereche cu metoda **destruct**). In plus, daca se doreste redeumirea clasei, acest lucru nu va afecta codul interior – constructorul nu va trebui modificat.
Spre deosebire de PHP 4, unde obiectele erau transmise prin valoare, in **PHP 5**, acestea sunt transmise prin referinta. De asemenea, vizibilitatea in interiorul unei clase prezinta cateva particularitati; Exemplul urmator arata modalitatea corecta de accesare a membrilor unei clase:

```
class Example {
   private $id;
   public $string;
   public function __construct($id, $string)
   {
       $this->id = $id;
       $this->string = $string;
   }
}
```

Membrii unei clase nu se pot apela direct; spre exemplu folosirea lui **$string** in interiorul constructorului de mai sus este echivalenta cu declararea unei variabile locale cu numele **$string**, diferita de membrul clasei **$this→string**;
In PHP 5 se pot defini si apela metode statice astfel:

```
class Example {
     public static function foo () {
     }
}
Example::foo(); // apelul unei metode statice
```

Search

Table of Contents

Laborator 02 - BD in PHP
Clase in PHP
Conectarea la baza de date
Prepared statements
Query-uri simple
Exercitii
Discussion

- Repartizare teme
- Catalog PW

Laboratoare

- Laborator 01 - Introducere
- Laborator 02 - BD in PHP
- Laborator 03 - Arrays, Magic Methods
- Laborator 04 - (X)HTML, CSS
- Laborator 05 - Formulare HTML, Persistența Datelor
- Laborator 06 - Securitate I
- Laborator 07 - Securitate II
- Laborator 08 - JavaScript
- Laborator 09 - DOM scripting
- Laborator 10 - AJAX
- Laborator 11 - Web Frameworks

Teme

- Tema 01 - API pentru BD
- Tema 02 - Template System & Controller
- Tema 03 - Generator de formulare
- Tema 04 - Tree Web UI

Login

Conectarea la baza de date

Prepared statements

Pentru conectarea la baza de date MySQL, in PHP 5 pune la dispozitie clasele mysqli si mysqli-stmt. Exemplul urmator arata modalitatea de conectare
la o baza de date. Presupunem ca baza de date are un tabel cu urmatoarea structura:

Tabelul users		
id	**username**	**description**
1	mihai	Mihai's description

```
$id = 1;
$connection = new mysqli('localhost','username','password','database-name'); // linia 2
$query = "SELECT * from users where id = ?"; //linia 3
$stmt = $connection->prepare($query); //linia 4
$stmt->bind_param("i",$id); //linia 5
$stmt->execute(); //linia 6
$stmt->bind_result($id,$username,$description); //linia 7
$stmt->fetch(); // linia 8
$stmt->close();
$connection->close();
echo $id;
echo $username;
echo $description;
```

La **linia 2**, se creeaza un obiect mysqli - conexiunea la baza de date. Parametrii uzuali sunt adresa (folosim deseori localhost, intrucat baza de date se afla local in raport cu serverul care ruleaza codul PHP), username-ul si parola asociate unui utilizator al bazei de date (crearea unui utilizator si adaugarea de drepturi acestuia pentru accesul la o baza de date, se realizeaza din interfata de administrare MySQL).
Linia 3 formateaza un query catre baza de date. Se observa ca acest query contine (unul sau mai multe) simboluri ?. Acestea vor fi inlocuite cu valori de variabile, dupa cum vedem in continuare;
Linia 4 creeaza un prepared statement;
Linia 5 atribuie prepared statement-ului creat, valorile variabilelor. Mecanismul este urmatorul: primul parametru contine un string cu unul sau mai multe caractere. Fiecare caracter desemneaza tipul variabilei de inlocuit. Urmatorii parametrii (in exemplul nostru exista doar unul) sunt variabilele ce vor fi "inlocuite" cu ?.
Altfel spus, apelul din **linia 5**, inlocuieste simbolul ? din query cu o variabila de tip integer (de unde "i"), al carui continut este $id.
Linia 6 executa query-ul obtinut prin inlocuirea variabilelor.
Linia 7 asociaza campurile intoarse cu cele trei variabile - parametrii
Linia 8 executa preluarea efectiva a datelor (din acest moment, cele trei variabile contin valorile campurilor din baza de date)
Observatii:
- cand executarea unui query produce mai multe rezultate, apelul fetchva intoarce doar pe primul dintre ele; prin apeluri succesive ale fetch (pana cand nu mai exista rezultate), se pot obtine toate rezultatele unui query;
- desi prepared statement-urile par dificil de folosit la prima vedere, ele sunt utile in situatii in care dorim executarea de query-uri generice; ele sunt de asemenea utile in a garanta corectitudinea codului scris;

Query-uri simple

```
$connection = new mysqli('localhost','username','password','database-name'); //linia 2
$query = "SELECT * from users where id = 1"; //linia 3
$result = $connection->query($query); //linia 4
while ($row = $result->fetch_assoc() ) //linia 5
{
   echo $row["userId"];
}
```

La **linia 2**, se stabileste o conexiune, similar cu exemplul anterior;
Linia 3 creeaza un query simplu;
Linia 4 executa query-ul, si obtine un obiect result, ce contine rezultatele executarii query-ului;
Linia 5 parcurge inregistrare cu inregistrare, rezultatul intors. Metoda fetch_assoc intoarce un array asociativ ce contine drept chei numele coloanelor din tabel, iar drept valori, valoarile inregistrarii curente, asa cum se poate observa la linia 7.

Exercitii

- Fie urmatoarea interfata:

IUser.interface.php

```
define ("DB", "pwdatabase1");
define ("USER", "root");
define ("PASS", "");
```

```
interface IUser {
   public function __construct ($id);
   public function __toString ();
}
```

- Scrieti o clasa (`User`) care implementeaza interfata IUser:
  - (2p) Metoda `construct` primeste un id, si extrage informatiile asociate cu acel id, din baza de date;
  - (1p) Metoda `toString` intoarce o reprezentare sub forma de sir de caractere a Utilizatorului (ex: **[1,Mihai,Mihai's description]**);
  - (1p) Instantiati clasa voastra, si incercati sa afisati direct instanta (ex: **`echo $instance;`**); Cum explicati comportamentul ?
- (2p) Adaugati clasei voastre o metoda (`update`) care permite modificarea parametriilor unui utilizator (nume, descriere); Testati;
- (2p) Adaugati clasei voastre o metoda (`insert`) care primeste campurile tabelului, si creeaza o noua inregistrare in baza de date; Testati;
- (1p) Adaugati clasei **`User`** o metoda statica ce cauta un utilizator dupa prenume, si il intoarce pe primul gasit. Metoda statica va intoarce un obiect de tip User.
- (1p) Creati o noua clasa (`MyException`) care extinde clasa PHP **`Exception`**; Supradefiniti membrul **`protected $message`**; Atribuiti-i un mesaj corespunzator;
- (1p) Modificati constructorul clasei, astfel incat, daca id-ul are o valore incorecta, acesta va arunca o exceptie de tip **`MyException`**; Incercati sa prindeti exceptia in codul vostru de testare;
- **(Bonus 3p)** Scrieti o noua clasa MyObject (dupa modelul lui User) care realizeaza aceleasi operatii ca si User (extrage, updateaza si insereaza) insa pe o inregistrare generica (tabel cu orice structura, care insa contine un index 'id'). Folositi query-uri simple.
- **Tips:**
  - Pentru a obtine id-ul ultimei inregistrari inserate intr-un tabel, folositi:
    - **`$stmt->insert_id;`** unde **`$stmt`** este un obiect **`mysqli_stmt`** (prepared statement)
    - **`$connection->insert_id;`** unde **`$connection`** este un obiect **`mysqli`**
  - Pentru includerea unui fisier extern script-ului curent folositi **`require_once('nume-fisier')`**
  - Detalii despre exceptii puteti gasi la: http://php.net/manual/en/language.exceptions.php
  - Consultati http://www.php.net pentru informatii complete privind comportamentul functiilor descrise in laborator.

Show pagesource Old revisions Back to top