PDO

PHP Data Object

Principes

- extension d'accès aux bases de données
- Compatible avec de nombreux SGBR
- Interface orientée objet
- DBAL (DataBase Access Layer)

SGBDR

pilote	SGBDR
PDO_MYSQL	MySQL
PDO_PGSQL	PostgreSQL
PDO_SQLITE	SQLite
PDO_IBM	IBM DB2
PDO_OCI	Oracle
PDO_DBLIB	MS SQL Server, Sybase
PDO_FIREBIRD	Firebird, Interbase
PDO_ODBC	ODBC

Installation

Extension PHP doit être activée dans php.ini (supprimer le point-virgule et relancer le serveur Apache)

```
;extension=php_pdo_firebird.dll
extension=php_pdo_mysql.dll
;extension=php_pdo_oci.dll
;extension=php_pdo_odbc.dll
;extension=php_pdo_pgsql.dll
;extension=php_pdo_sqlite.dll
```

Connexion

pilote

adresse du serveur

nom de la BD

```
$dsn = 'mysql:host=localhost;
        dbname=maBase;charset=utf8';
$user = 'root';
$pass = '';

$connexion = new PDO($dsn, $user, $pass);
```

Éléments du DSN

élément	description
pilote	préfixe du DSN indique le pilote à utiliser ex : mysql:
host	adresse du serveur (sous linux, une connexion utilisant localhost passera forcément par un socket)
port	port écouté par le serveur (si différent du port standard)
dbname	nom de la base de données
unix_socket	le socket utilisé (remplace host et port, uniquement utilisable sous unix ou linux si le SGBDR et le serveur web sont sur le même serveur physique)
charset	le jeu de caractère de la connexion (à partir de PHP 5.3.6)

Gestion des erreurs

```
$dsn = 'mysql:host=localhost;
       dbname=maBase;charset=utf8';
$user = 'root';
$pass = '';
$options = [
   PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
];

$connexion = new PDO($dsn, $user, $pass, $options);
```

Fonction de connexion

```
define('DSN', 'mysql:host=localhost;dbname=maBase');
define('DB_USER', 'root');
define('DB_PASS, '');
```

config.php

```
function getPDO(){
   $options = [
      PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
   ];

   return new PDO(DSN,DB_USER,DB_PASS, $options);
}
```

libPDO.php

Requêtes

Exécution d'une requête SELECT

```
$connexion = getpDO();
$sql = "SELECT * FROM clients";
$resultSet = $connexion->query($sql);
```

Récupération des données

méthodes fetch

récupère une ligne et place le curseur à la ligne suivante

```
while($row = $resultSet->fetch(PDO::FETCH_ASSOC)){
   echo $row['nom'].'<br>';
}

while($row = $resultSet->fetch(PDO::FETCH_NUM)){
   echo $row[0].'<br>';
}

while($row = $resultSet->fetch(PDO::FETCH_OBJ)){
   echo $row->nom.'<br>';
}
```

Récupération des données

méthodes fetchALL

récupère toutes les lignes de la requête dans un tableau ordinal

```
$rows = $resultSet->fecthAll(PDO::FETCH_ASSOC);
$resultSet = null;
$nbRows = count($rows);

for($i=0; $i < $nbRows; $i++){
   echo $rows[$i]['nom'].'<br>';
}
```

Modes de récupération

PDO::FETCH_NUM	récupère un tableau ordinal de valeurs
PDO::FETCH_ASSOC	récupère un tableau associatif de valeurs
PDO::FETCH_BOTH	combine FETCH_NUM et FETCH_ASSOC
PDO::FETCH_OBJ	récupère un objet anonyme dont les propriétés sont les colonnes de la requête
PDO::FETCH_LAZY	combine FETCH_OBJ et FETCH_BOTH

Exécution

Toute requête hors SELECT

```php
$connexion = getpDO();
$sql = "DELETE FROM clients WHERE id=1";

//exécute la requête et retourne
//le nombre de lignes affectées
$affectedRecords = $connexion->exec($sql);
```

Dernier id auto

```
$connexion = getpDO();
$sql =    "INSERT INTO clients (nom, prenom)
          VALUES ('MARTIN', 'Pierre')";

$affectedRecords = $connexion->exec($sql);

//récupère et affiche le dernier numéro automatique
$id = $connexion->lastInsertId();
```

Requêtes préparées

Principes

- Une requête utilisant des marqueurs à la place des données
- La requête est compilée, le SGBDR établit un plan d'exécution
- A chaque exécution il est possible d'injecter des nouvelles données en conservant le même plan d'exécution.

pourquoi

- Performance : requête exécutée plusieurs fois avec des données différentes
- Sécurité : injection SQL impossible

Requête préparée

marqueurs ordinaux

```
$connexion = getpDO();

$sql =    "UPDATE clients SET nom=? WHERE id=?";
$statement = $connexion->prepare($sql);

$statement->execute(['Martin', 1]);

$statement->execute(['Ferrier', 3]);

$statement->execute(['Meunier', 5]);
```

Requête préparée

marqueurs associatifs

```
$connexion = getpDO();

$sql =    "UPDATE clients
          SET nom=:nom WHERE id=:id";

$statement = $connexion->prepare($sql);

$statement->execute(['nom'=>Martin','id'=>1]);

$statement->execute(['nom'=>'Ferrier','id'=>3]);

$statement->execute(['nom'=>'Meunier','id'=>5]);
```

Les marqueurs ne remplacent que les valeurs (pas les noms de colonne ou de table)

Paramètres typés

marqueurs ordinaux

```
$connexion = getpDO();

$sql =       "SELECT * FROM clients WHERE
             code_postal=? LIMIT ?";

$statement = $connexion->prepare($sql);

$statement->bindValue(1, '25000', PDO::PARAM_STR);
$statement->bindValue(2, 5, PDO::PARAM_INT);

$rs = $statement->execute();

$clients = $rs->fetchAll(PDO::FETCH_ASSOC);
```

Paramètres typés

marqueurs associatifs

```
$connexion = getpDO();

$sql =      "SELECT * FROM clients WHERE
            code_postal=:cp LIMIT :limit";

$statement = $connexion->prepare($sql);

$statement->bindValue('cp','25000',PDO::PARAM_STR);
$statement->bindValue('limit',5,PDO::PARAM_INT);

$rs = $statement->execute();

$clients = $rs->fetchAll(PDO::FETCH_ASSOC);
```

Paramètres liés aux variables

```
$connexion = getpDO();

$limit = 5;

$sql = "SELECT * FROM clients LIMIT :limit";

$statement = $connexion->prepare($sql);

$statement->bindParam(  'limit',
                        $limit, PDO::PARAM_INT);

$rs = $statement->execute();

$clients = $rs->fetchAll(PDO::FETCH_ASSOC);
```

Résultat lié aux variables

```
$connexion = getpDO();
$sql = "SELECT id, nom FROM auteurs";
$statement = $connexion->prepare($sql);

$statement->bindColumn(1, $id, PDO::PARAM_INT);
$statement->bindColumn(2, $nom, PDO::PARAM_STR);

$statement->execute();

while($statement->fetch(PDO::FETCH_BOUND)){
    echo "$id : $nom <br>";
}
```

Gestion des erreurs

Exceptions

```
try {

   $connexion = getpDO();
   $sql = "SELECT id, nom FROM auteurs";
   $rs = $connexion->query($sql);
   $rows = $rs->fetchAll(PDO::FETCH_ASSOC);

   …

} catch(PDOException $e){
   echo $e->getMessage();
}
```

Transactions

Principes

Plusieurs actions sont liées, si l'une d'elles échoue, l'ensemble des opérations est annulée

Transactions

```php
$connexion = getpDO();
$connexion->beginTransaction();

try {
    $sql = "INSERT INTO auteurs (nom,prenom) VALUES (?,?)";
    $stm = $connexion->prepare($sql);
    $stm->execute(['Auster', 'Paul']);

    $authorId = $connexion->lastInsertId();

    $sql = "INSERT INTO livres (titre, auteur_id) VALUES (?,?)";
    $stm = $connexion->prepare($sql);
    $stm->execute(['Leviathan', $authorId]);

    $connexion->commit();

} catch (PDOException $e){
    $connexion->rollBack();
    echo $e->getMessage();
}
```