Accéder à une base de données avec PDO

Application au SGBD MySQL

L’accès au SGBD en PHP

PHP s'interface nativement avec un grand nombre de SGBD dont MySQL par excellence
Au besoin, recours à des *middleware* (ODBC, **PDO**…) pour communiquer avec tout SGBD
L'interfaçage se fait au moyen de bibliothèques de fonctions spécifiques à chaque SGBD (*extensions* PHP), à l'aide d'instanciation d'objets (MySQLi) ou à l'aide d'un Framework d'accès aux BDD (**PDO**)

Les fonctions incontournables consistent à :

- se connecter **au moteur**
- **préciser la** base cible
- soumettre la requête SQL **à exécuter**
- récupérer le résultat **pour le traiter et renvoyer au client un flot d’informations HTML**
- se déconnecter

PHP et MySQL

MySQL est un SGBD très prisé dans les environnements Linux/Apache

MySQL est très présent entre autres sur les sites d'hébergement

PHP offre 3 moyens d'accéder à une base de données **MySQL** :

- Jeu d'instructions **mysql_xxx** historique mais déprécié (extension PHP)

- Jeu d'instructions **mysqli_xxx** (extension PHP) et les objets **MySQLi** instanciables en PHP (maintenant déprécié)

- Framework orienté objet PDO qui permet d'accéder à divers SGBD dont MySQL

La Classe PDO

PDO est un Framework (=ensemble de classes) destiné à prendre en charge toute la "*quincaillerie*" nécessaire pour accéder en PHP à une base de données

- Permet de faire abstraction du SGBD réellement mis en œuvre

- Favorise l'évolution de l'application

- Orienté objet

- Supporte les requêtes ‘**préparées**’ qui réalisent automatiquement les concaténations périlleuses des libellés de requêtes SQL (permet de lutter contre les **injections SQL**)

Se connecter à une Base MySQL

L'essentiel :

```php
// Ouverture d'une connexion sur la Base magasin du SGBD MySQL
$dsn = "mysql:dbname=magasin;host=localhost:3308";
$option = array(PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
                PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8');
$connexion = new PDO($dsn, "root", "", $option);
```

Le premier paramètre permet de préciser les modalités de fonctionnement en cas d'erreur à la connexion, il permet une levée d'exception :
PDO::ATTR_ERRMODE=> PDO::ERRMODE_EXCEPTION

Le deuxième paramètre permet de gérer les problèmes d'accentuation entre le script PHP et le SGBD
PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8'
Il peut être remplacé par le paramètre **charset=utf8** dans l'objet **$dsn**.

Se connecter à une Base MySQL …

Ou encore avec gestion d'erreur :

```php
// Ouverture d'une connexion sur la Base magasin du SGBD MySQL
$dsn = "mysql:dbname=magasin;host=localhost:3308";
try {
    $option = array(PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
                    PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8');
    $connexion = new PDO($dsn, "root", "", $option);

} catch (PDOException $e) {
    printf("Echec connexion : %s\n", $e->getMessage());
}
```

PDO : Extraction de données (requête SQL Select)

L'essentiel :

```php
$reponse = $connexion->query("select * from article");
// Affichage de la liste des articles
while($donnees = $reponse->fetch()){

    echo $donnees["designation"]."<br>";
}
// Fermer le curseur d'analyse des résultats. A faire à chaque fois que vous avez
// terminé de traiter le retour d'une requête.
$reponse->closeCursor();
```

Ou :

```php
// Récupération des articles dans la Table article
$reponse = $connexion->query("select * from article");
foreach($reponse as $ligne){
    echo $ligne["designation"]."<br>";
}
```

Ou encore :

```php
// Récupération des articles dans la Table article
$reponse = $connexion->query("select * from article");
$records=$reponse->fetchAll(PDO::FETCH_COLUMN,1);
foreach($records as $ligne){
    echo $ligne."<br>";
}
```

Liste des Articles

Canon EOS 3000V zoom 28/80
Cassette DV60 par 5
Camescope Panasonic SV-AV 100
Caméscope Sony DCR-PC330
Portable Dell X300
DVD vierge par 3
PC Bureau HP497 écran TFT
Nikon F55+zoom 28/80
Nikon F80
Portable Samsung X15 XVM
PC Portable Sony Z1-XMP

PDO : Extraction de données (requête SQL Select)

```php
<?php
// Ouverture d'une connexion sur la Base magasin du SGBD MySQL
$dsn = "mysql:dbname=magasin;host=localhost:3308";
try {
    $option = array(PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
                    PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8');

    $connexion = new PDO($dsn, "root", "", $option);

} catch (PDOException $e) {
    printf("Echec connexion : %s\n", $e->getMessage());
}
// Récupération des articles dans la Table article
$reponse = $connexion->query("select * from article");
?>
<body>
    <h1>Liste des Articles</h1>
    <?php
    // Affichage de la liste des articles
    while($donnees = $reponse->fetch()){

        echo $donnees[1]."<br>";
    }
    // Fermer le curseur d'analyse des résultats. A faire à chaque fois que vous avez
    // terminé de traiter le retour d'une requête.
    $reponse->closeCursor();
    ?>
</body>
</html>
```

Liste des Articles

Canon EOS 3000V zoom 28/80
Cassette DV60 par 5
Camescope Panasonic SV-AV 100
Caméscope Sony DCR-PC330
Portable Dell X300
DVD vierge par 3
PC Bureau HP497 écran TFT
Nikon F55+zoom 28/80
Nikon F80
Portable Samsung X15 XVM
PC Portable Sony Z1-XMP

PDO : Récupérer le résultat d'une requête SQL Select

PDO retourne les données sous forme de :

- Tableau PHP indicé : $data=$resultats->fetch(**PDO::FETCH_NUM**);

```php
// Affichage de la liste des articles
while($donnees = $reponse->fetch(PDO::FETCH_NUM)){

    echo $donnees[1]."<br>";
}
```

- Tableau PHP associatif : $data=$resultats->fetch(**PDO::FETCH_ASSOC**);

```php
// Affichage de la liste des articles
while($donnees = $reponse->fetch(PDO::FETCH_ASSOC)){

    echo $donnees["designation"]."<br>";
}
```

- Objet standard PHP : $data=$resultats->fetch(**PDO::FETCH_OBJ**);

```php
// Affichage de la liste des articles
while($donnees = $reponse->fetch(PDO::FETCH_OBJ)){

    echo $donnees->designation."<br>";
}
```

- **Par défaut, Tableau PHP associatif et indicé** : $data=$resultats->**fetch()**; qui correspond à la constante de classe **PDO::FETCH_BOTH**

PDO : Les requêtes préparées SELECT

Technique pour réaliser des requêtes SQL paramétrées avec le ou les **?** qui sera ou seront alimentés par un tableau de **un** ou **plusieurs** paramètres transmis par la méthode execute() de l'objet d'accès au jeu d'enregistrements $reponse

- **PDO** réalise les concaténations périlleuses du libellé de requête SQL

```
24  // Récupération des articles dans la Table article appartenant à la catégorie 'photo'
25  $sql = "select * from article where categorie= ?";
26  $reponse = $connexion->prepare($sql);
27  $reponse->execute( array("photo"));
28  ?>
29  <body>
30      <h1>Liste des Articles de la Catégorie Photo</h1>
31      <?php
32      // Affichage de la liste des articles de la catégorie 'photo'
33      while($data = $reponse->fetch(PDO::FETCH_BOTH)){
34
35          echo $data["designation"]."<br>";
36      }
37      // Fermer le curseur d'analyse des résultats. A faire à chaque fois que vous avez
38      // terminé de traiter le retour d'une requête.
39      $reponse->closeCursor();
40      ?>
41  </body>
42  </html>
```

Liste des Articles de la Catégorie Photo

Canon EOS 3000V zoom 28/80
Nikon F55+zoom 28/80
Nikon F80

PDO : Les requêtes préparées SELECT …

Technique pour réaliser des requêtes SQL **paramétrées** mais cette fois avec des **paramètres nommés, les étiquettes ou marqueurs nominatifs (:cat, :design)**. Cette fois on peut passer les valeurs affectées à ces paramètres dans **n'importe quel ordre** avec un **tableau associatif** transmis à l'objet

```
24  // Récupération des articles dans la Table article appartenant à la catégorie 'video'
25  // et dont la désignation comporte le mot 'Camescope'
26  $sql = "select * from article where categorie = :cat and designation like :design ";
27  $reponse = $connexion->prepare($sql);
28  $reponse->execute( array(":cat"=>"video", ":design"=>"%Camescope%"));
29  ?>
30  <body>
31      <h1>Liste des Articles </h1>
32      <?php
33      // Affichage de la liste des articles de la catégorie 'photo'
34      while($data = $reponse->fetch(PDO::FETCH_BOTH)){
35
36          echo $data["designation"]."<br>";
37      }
38      // Fermer le curseur d'analyse des résultats. A faire à chaque fois que vous avez
39      // terminé de traiter le retour d'une requête.
40      $reponse->closeCursor();
41      ?>
42  </body>
43  </html>
```

Liste des Articles

Camescope Panasonic SV-AV 100
Caméscope Sony DCR-PC330

PDO : Les requêtes préparées INSERT


```php
<?php
    // Ouverture d'une connexion sur la Base magasin du SGBD MySQL
    $dsn = "mysql:dbname=magasin;host=localhost:3308";
    try {
        $option = array(PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
                        PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8');

        $connexion = new PDO($dsn, "root", "", $option);

    } catch (PDOException $e) {
        printf("Echec connexion : %s\n", $e->getMessage());
    }

    // Préparation de la requête avec des marqueurs nommés
    $sql = "insert into article values (:code, :designation, :prix, :categorie)";
    $reponse = $connexion->prepare($sql);

    // Récupération des valeurs issues de la soumission du formulaire
    $code           = $_POST["code"];
    $designation    = $_POST["designation"];
    $prix           = $_POST["prix"];
    $categorie      = $_POST["categorie"];

    // Exécution de la requête préparée d'insertion sans contrôle de validation
    $reponse->execute( array(   ":code"=>$code,
                                ":designation"=>$designation,
                                ":prix"=>$prix,
                                ":categorie"=>$categorie));
?>
```

PDO : Les requêtes préparées INSERT Sécurisées

```php
    // Préparation de la requête avec des marqueurs nommés
    $sql = "insert into article values (:code, :designation, :prix, :categorie)";
    $reponse = $connexion->prepare($sql);

    // Récupération des valeurs issues de la soumission du formulaire
    $code           = $_POST["code"];
    $designation    = $_POST["designation"];
    $prix           = $_POST["prix"];
    $categorie      = $_POST["categorie"];

    // Exécution de la requête préparée d'insertion sans contrôle de validation
    // $reponse->execute( array(   ":code"=>$code,
    //                             ":designation"=>$designation,
    //                             ":prix"=>$prix,
    //                             ":categorie"=>$categorie));

    // Version de l'insertion avec bindValue (on peut passer des valeurs aux différents marqueurs)
    // De plus on peut sécuriser les données à insérer en utilisant les constantes de classe PDO
    // PDO::PARAM_STR, PDO::PARAM_INT, PDO::PARAM_BOOL

    $reponse->bindValue(":code", $code, PDO::PARAM_STR);
    $reponse->bindValue(":designation", $designation, PDO::PARAM_STR);
    $reponse->bindValue(":prix", $prix, PDO::PARAM_STR);
    $reponse->bindValue(":categorie", $categorie, PDO::PARAM_STR);
    $reponse->execute();
?>
```

La fonction **bindValue()** permet **d'affecter** les valeurs des variables aux marqueurs et de les **tester** avant l'insertion

Lorsqu'il s'agit de sécuriser les données à insérer dans une table, on peut utiliser les **constantes de classe PDO**.

PDO::PARAM_STR , on demande à PDO de ne laisser passer que les types de données VARCHAR et CHAR

PDO::PARAM_INT, on demande à PDO de ne laisser passer que les entiers

PDO::PARAM_BOOL, on demande à PDO de ne laisser passer que les booléens

PDO : Les requêtes préparées UPDATE


```php
<?php
    // Ouverture d'une connexion sur la Base magasin du SGBD MySQL
    $dsn = "mysql:dbname=magasin;host=localhost:3308";
    try {
        $option = array(PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
                        PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8');

        $connexion = new PDO($dsn, "root", "", $option);

    } catch (PDOException $e) {
        printf("Echec connexion : %s\n", $e->getMessage());
    }

    // Préparation de la requête avec des marqueurs nommés
    $sql = "update article set designation=:designation, prix=:prix, categorie=:categorie where id_article=:code";

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

    // Récupération des valeurs issues de la soumission du formulaire
    $designation    = $_POST["designation"];
    $prix           = $_POST["prix"];
    $categorie      = $_POST["categorie"];
    $code           = $_POST["code"];

    // Exécution de la requête préparée de modification sans contrôle de validation
    $reponse->execute( array(   ":designation"=>$designation,
                                ":prix"=>$prix,
                                ":categorie"=>$categorie,
                                ":code"=>$code));

    header("location:indexupdate.php");
?>
```

PDO : Les requêtes préparées UPDATE Sécurisées

```php
    // Préparation de la requête avec des marqueurs nommés
    $sql = "update article set designation=:designation, prix=:prix, categorie=:categorie where id_article=:code";

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

    // Récupération des valeurs issues de la soumission du formulaire
    $designation    = $_POST["designation"];
    $prix           = $_POST["prix"];
    $categorie      = $_POST["categorie"];
    $code           = $_POST["code"];

    // Exécution de la requête préparée de modification sans contrôle de validation
    // $reponse->execute( array(    ":designation"=>$designation,
    //                              ":prix"=>$prix,
    //                              ":categorie"=>$categorie,
    //                              ":code"=>$code));

    $reponse->bindValue(":designation", $designation, PDO::PARAM_STR);
    $reponse->bindValue(":prix", $prix, PDO::PARAM_STR);
    $reponse->bindValue(":categorie", $categorie, PDO::PARAM_STR);
    $reponse->bindValue(":code", $code, PDO::PARAM_STR);
    $reponse->execute();

    header("location:indexupdate.php");
?>
```

Comme pour l'insertion on peut utiliser la fonction **bindValue()** pour affecter les valeurs des variables aux marqueurs et puis les tester.

Vous rencontrerez aussi la fonction **bindParam()** qui fait le même travail que la fonction **bindValue()** mais qui peut lier plusieurs valeurs aux même variables. En fait, une variable est liée par référence à un marqueur donné.

PDO : Les requêtes préparées DELETE

Suppression d'un Article

Code	Désignation	Prix	Catégorie	Suppression
CA300	Canon EOS 3000V zoom 28/80	100.00	photo	Supprimer
CAS07	Cassette DV60 par 5	45.00	divers	Supprimer
CP100	Camescope Panasonic SV-AV 100	1490.00	video	Supprimer
CS330	Caméscope Sony DCR-PC330	1629.00	video	Supprimer
DEL30	Portable Dell X300	1715.00	informatique	Supprimer

```php
<?php
    // Ouverture d'une connexion sur la Base magasin du SGBD MySQL
    $dsn = "mysql:dbname=magasin;host=localhost:3308";
    try {
        $option = array(PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
                        PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8');

        $connexion = new PDO($dsn, "root", "", $option);

    } catch (PDOException $e) {
        printf("Echec connexion : %s\n", $e->getMessage());
    }

    // Préparation de la requête de suppression avec le marqueurs nommé :code
    $sql = "delete from article where id_article=:code";
    $reponse = $connexion->prepare($sql);

    // Récupération du code passé en GET à partir du lien hypertexte
    $code= $_GET["code"];

    // Exécution de la requête préparée de suppression
    $reponse->execute( array(":code"=>$code));

    // Retour à la liste des articles
    header("location:indexdelete.php");
?>
```

PDO : Pour aller plus loin …

1 - Paramètre supplémentaire à l'initialisation de l'objet PDO :
Pour gérer les caractères accentués des variables à insérer ou à modifier dans les tables on peut ajouter dans la variables **$dsn** le paramètre : **charset=utf8**

```php
// Ouverture d'une connexion sur la Base magasin du SGBD MySQL
$dsn = "mysql:dbname=magasin;host=localhost:3308;charset=utf8";
try {
    $option = array(
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION);

    $connexion = new PDO($dsn, "root", "", $option);
} catch (PDOException $e) {
    printf("Echec connexion : %s\n", $e->getMessage());
}
```

2 – Récupérer le dernier enregistrement ajouté :
Après une **nouvelle insertion** dans une table, il peut être parfois utile de pouvoir connaître la valeur de la clé primaire auto incrémentée du dernier enregistrement effectué : Pour cela on peut utiliser la méthode **lastInsertId()** qui appartient à l'objet connexion.

```php
$connexion->lastInsertId();
```

3 – Connaître le nombre d'enregistrement :
Pour afficher le nombre d'enregistrements dans une Table on peut utiliser la méthode **rowCount()**

```php
$reponse->rowCount()
```

CRÉDITS

OEUVRE COLLECTIVE DE L'AFPA

Sous le pilotage de la DIIP
et du centre sectoriel Tertiaire

EQUIPE DE CONCEPTION

M. Restoueix Sacha (Formateur)

Date de mise à jour : 28/12/2020
Date de dépôt légal : 2020
