Exposé de Projet BD

Thème: Manipulation et Connexion à une Base de données en PHP et utilisations des Frameworks Zend Framework et CodeIgniter

EXPOSANTS:

- YOBA ROSTAND
- KAMGA II JAIME
- MBIDA MARC
- NDOMO SONIA
- MASSAGA ARISTIDE

Sous la supervision du

Dr, Ing Nana Mbinkeu

PLAN DE L’EXPOSE

Introduction

Comment Se connecter à une BD en PHP?

Connexion à une BD avec PDO

Un peu d’histoire sur les Framework PHP

Utilisation de Zend FrameWork

Utilisation de CodeIgniter

Conclusion

INTRODUCTION

PHP (***Hypertext Preprocessor***) est un langage de scripts libre et orienté objet depuis sa version 5, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP. L'une des fonctionnalités que offre le PHP et qui fait l'objet de notre exposé aujourd'hui est la connexion aux Bases de Données ici **MySql**, par **PHP** puis via des **Framework** comme **Zend** et **CodeIgniter**.

Comment se connecte-t-on à la base de données en PHP ?

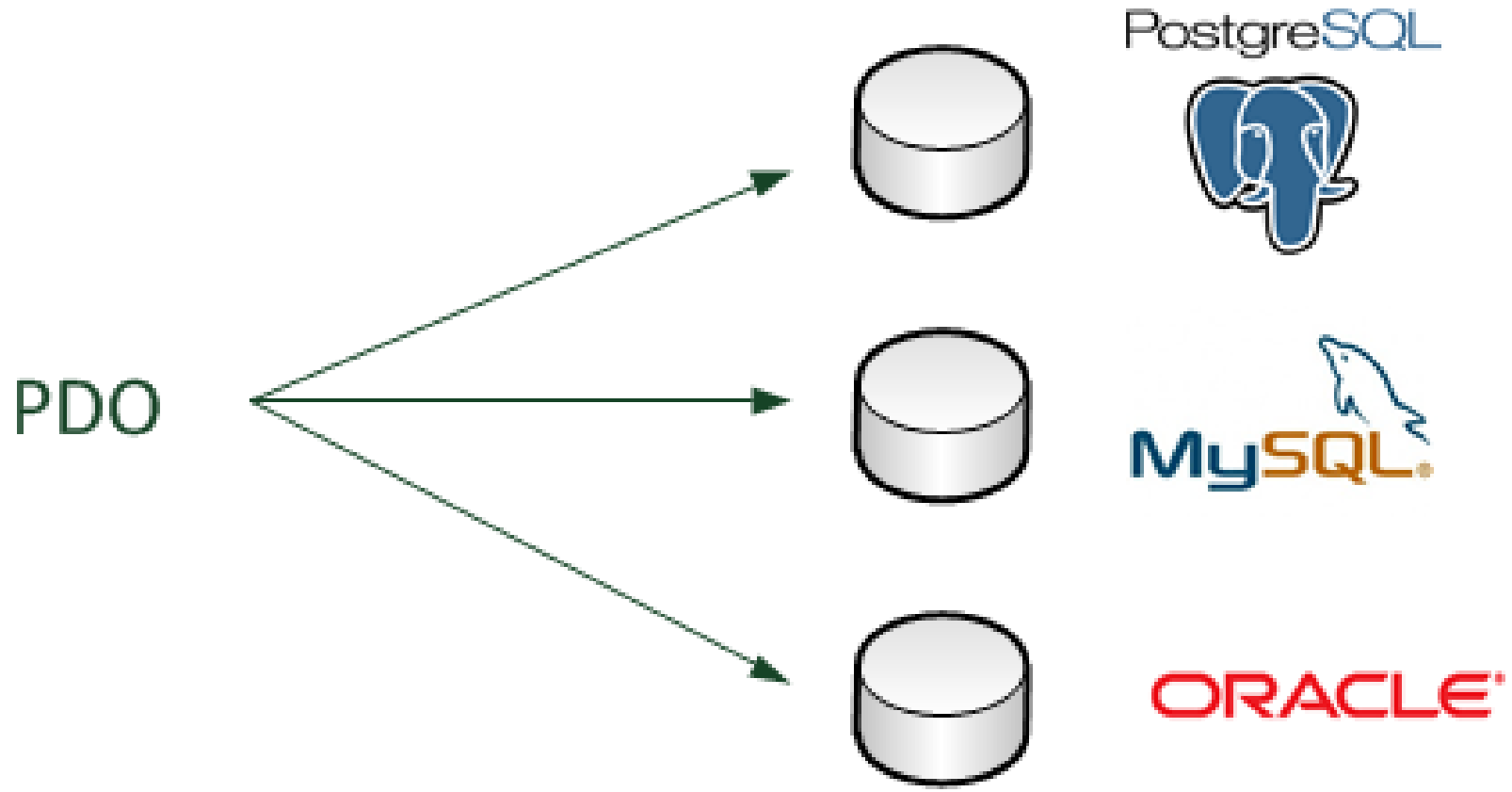
En effet, PHP propose plusieurs moyens de se connecter à une base de données MySQL. Il met à notre disposition plusieurs extensions à savoir

- L'extension **mysql_** : ce sont des fonctions qui permettent d'accéder à une base de données MySQL et donc de communiquer avec MySQL. Leur nom commence toujours par mysql_. Toutefois, ces fonctions sont vieilles et on recommande de ne les utiliser aujourd'hui.
- L'extension **mysqli_**: ce sont des fonctions améliorées d'accès à MySQL. Elles proposent plus de fonctionnalités et sont plus à jour.
- L'extension **PDO(Php Data Objet)**: c'est l'outil complet qui permet d'accéder à n'importe quel type de base de données. On peut donc l'utiliser pour se connecter aussi bien à Mysql que PostgreSQL ou Oracle.

PDO

Tous ces éléments sont des extensions car PHP est très modulaire.

Dans cet exposé vu l'utilisation avancé et l'ouverture que nous donne PDO dans cet exposé nous présenterons PDO : PHP Data Objet.



Connexion à une Base de données avec PHP : Utilisation de PDO

Pour ce connecter à la base de données, il faut préciser 4 champs:

- **le nom de l'hôte** : c'est l'adresse de l'ordinateur où MySQL est installé (comme une adresse IP). Le plus souvent, MySQL est installé sur le même ordinateur que PHP : dans ce cas, mettez la valeur localhost (cela signifie « sur le même ordinateur »).
- **la base** : c'est le nom de la base de données à laquelle vous voulez vous connecter. Dans notre cas, la base s'appelle test.
- **le login** : il permet de vous identifier.
- **le mot de passe**

Exemple de code

```
<?php
  $bdd = new PDO('mysql:host=localhost;dbname=test', 'root', "");
catch(Exception $e)
        {
  die('Erreur : '.$e->getMessage());
   }
?>
```

Apres avoir créée la base de données, on s'interresse sur comment insérer les tuples.

Dans ce cas, on utilisera la fonction exec() qui est prévue pour exécuter des modifications sur la base de données .

Comme exemple on le code:

```
$bdd->exec('INSERT INTO Etudiant(nom, prénom,matricule, niveau)
VALUES(\'MBIDA\',\'Marc\', \'12p175\', 3)');
```

Lorsqu'il s'agit des données variables,il est conseillé d'utiliser la fonction exec(), dont un exemple d'application est le suivant:

```
<?php
    $req = $bdd->prepare('INSERT INTO Etudiant(nom, prenom,
    matricule, niveau) VALUES(:nom,:prenom, :matricule, :niveau)');
$req->execute(array(
    'nom' => $nom,
    'prenom' => $prenom,
    'niveau' => $niveau,
    'matricule' => $matricule,
   ));
```

La fonction utilisée pour recuperer les tuples est la fonction query()

```
$reponse = $bdd->query('SELECT * FROM jeux_video');
while ($donnees = $reponse->fetch())
{
<strong>Nom de l’etudiant</strong> : <?php echo $donnees[‘nom']; ?><br />

}
$reponse->closeCursor(); // Termine le traitement de la requête
```

Pour conclure on peut dire que:

- Pour se connecter à une base de donnée, on a besoin de 4 champs dont nom de l’hote, base, le login et le mot de passe.
- On peut inserer les donnees dans la base de donnée grace aux fonctions exec() et prepare().
- L’execution des requettes s’effectue grace à la fonction exec(). Et que cette fonction renvoie plutôt un bloc de données, qu’on decompilera avec la fonction fetch().

Un Peu d'histoire sur les Frameworks PHP

En France, les principaux autres Framework que l'on trouve sur le marché des applications professionnelles sont les suivants :

• **Symfony** : un projet mûr qui propose une architecture solide, mais légèrement plus rigide. Il est appuyé par une grande communauté d'utilisateurs ainsi qu'une entreprise (Sensio).

• **Prado** : un framework sérieux qui propose une architecture intéressante et un fonctionnement

Très spécifique.

• **Copix** : un projet mûr à destination du monde professionnel, qui est capable de répondre à de

nombreux besoins.

• **Jelix** : un framework français, comme Copix, de bonne qualité.

• **CodeIgniter** : un framework de plus en plus populaire pour sa simplicité et ses performances.

La souplesse de Zend Framework est telle que, quelle que soit la base choisie, une collaboration cohérente peut être mise en place avec d'autres Framework ou composants.

Installation de Zend Framework

- **Téléchargement du paquet Zend Compressé.**

Télécharger la dernière version du paquet Zend à l'url suivante : http://framework.zend.com c'est l'adresse du site officiel.

Créer un dossier qu'on appellera Library dans C:/wamp/

Décompresser le fichier télécharger et copier le fichier « library » de zend framework dans le dossier C:/www/Library

- **Configuration du serveur apache**

Aller dans le fichier php.ini et modifier la directive include_path

include_path = ".;C:/www/Library/library"

Redémarrez apache et le tour est joué, le framework zend est installé et configuré.

Les SGBD utilisables par Zend Framework

Zend Framework propose le support des SGBD via PDO. (PDO : PHP Data Object)

Quelques un de ces SGBD sont:

MySql, Microsoft SQL Server, Oracle, PostgreSQL, SQLite, IBM DB2 et Informix Dynamic Server (IDS)

Manipulation de la base de données avec Zend Framework

Connexion à une Base de données avec Zend Framework

Avant de manipuler une base de données il faut d'abord avoir une connexion à la base de donnée. Pour créer une connexion à une base de données on utilise le code suivant :

```
<?php
$db = new Zend_Db_Adapter_Pdo_Mysql(array(
    'host '    => '127.0.0.1',
    'username' => 'root',
    'password' => '',
    'dbname'   => 'mysql'
));
```

La classe Zend_DB propose une méthode statique **factory()** qui peut faire exactement la même chose. Elle apporte cependant plus de flexibilité si l'on souhaite changer de SGBD dans le futur.

Manipulation de la base de données avec Zend Framework

Envoyer une requête à la database et récupérer

Pour envoyer une requête on utilise la méthode ***fetch*** chargée d'envoyer une requête et d'en récupérer les résultats. Pour plus de précision on peut utiliser les différentes sous méthodes fetch suivantes:

Sous méthodes fetch	Action effectué
fetchAll()	Récupère tous les résultats.
fetchRow()	Récupère le premier jeu de résultats.
fetchCol()	Récupère tous les résultats, mais uniquement la première colonne demandée.
fetchOne() = fetchRow()+fetchCol()	Retourne la première colonne du premier jeu de résultats.
fetchPairs()	Récupère le résultat sous forme de tableau associatif. La conne 1 est en index, la colonne 2 en résultat

Exemple de récupération de résultat suite à une requête SELECT

```
< ?php // Inclusion du composant Zend_Loader
include 'Zend/Db/Adapter/Pdo/Mysql.php';
include 'Zend/Debug.php';
$db = new Zend_Db_Adapter_Pdo_Mysql(array(
    'host'=> '127.0.0.1',
    'username'=> 'root',
    'password'=> '',
    'dbname'=>'zendframework'
));
$query = "SELECT * FROM user";
$result = $db->fetchAll($query);
Zend_Debug::dump($result);
```

Toutes les méthodes **fetch** prennent un paramètre de bind : il s'agit d'une chaine ou d'un tableau de chaînes à remplacer lors de la requête. Zend Framework utilise aussi les requêtes préparées en permanence

Exemple des user présent dans la BD à partir de leur id

```
$query = "SELECT *
        FROM user
        WHERE id=:id";
$tableau_id = range(1, 4);
foreach ($tableau_id as $id) {
    $binds = array('id'=>$id);
    $result = $db->fetchRow($query, $binds);
    echo "Nom = ".$result['nom'] . "  Prenom = " . $result['prenom']. "  matricule = ". $result['matricule'] . "  id = ".
$result['id'];
            echo "</br>";
            echo "</br>";
}
```

Insertion d’un élément dans une table de la base de données.

```
try {
    $data = array('nom' => 'nom a insere', 'prenom' => 'prenom a insere','matricule' =>
'12P200', 'id'=>'5');
    $count = $db->insert("user", $data);
            //Le count que vous avez ici est le nombre de lignes insérées dans la table user.
    echo $count . " user insere";
            }
catch (Zend_Db_Exception $e) {
    printf("erreur de requête : %s", $e->getMessage());
            }
```

Mise à jour de donnée dans la base de données

```
$updated = $db->update("user", array('nom' => "yoba update"), 'id=1');
echo $updated . " enregistrement(s) affecté";
```

Présentation du Framework CodeIgniter

Il a été conçu dans le but de ne fournir que le strict minimum. Tout le reste est entièrement optionnel (même les bibliothèques gérant les bases de données et les sessions le sont). CodeIgniter est une base réduite en fonctionnalités mais hautement performante, pouvant faire appel à des classes et à des fonctions très complètes lorsque le besoin s'en fait sentir .

Installation de CodeIgniter

Rendez-vous sur le site officiel de CodeIgniter *(http://codeigniter.com)* pour pouvoir télécharger le Framework et plus exactement dans la rubrique *downloads.*

- Une fois le Framework téléchargé, décompresser l'archive et placer le dossier portant le même nom que la version dans votre répertoire web *(Dans notre cas, c:\wamp\www)*

- Vous avez normalement accès à la page d'accueil du Framework *(Dans notre cas, http://localhost/codeIgniter) et celui-ci vous souhaite la bienvenue.*

Configuration de CodeIgniter

Le fichier config.php

Ce fichier de configuration est le cœur de la configuration du Framework. Il se trouve dans le dossier ./application/config/.

base_url

C'est l'url que vous devez taper pour accéder au fichier index.php.

```
/*
|--------------------------------------------------------------------
-------
| Base Site URL
|--------------------------------------------------------------------
-------
*/
$config['base_url'] = "http://localhost/codeIgniter/";
```

```
<?php

/*
|--------------------------------------------------------------------
-------
| Index File
|--------------------------------------------------------------------
-------
*/
// Dans le cas où mod_rewrite est activé
$config['index_page'] = "";

// Dans le cas contraire
$config['index_page'] = "index.php";
```

language

```php
<?php

/*
|-------------------------------------------------------------------
-------
| Default Language
|-------------------------------------------------------------------
-------
*/
$config['language'] = "french";
```

Session

```php
<?php

// Le nom du cookie...
$config['sess_cookie_name'] = 'ci_session';

// La date de péremption du cookie, en secondes...
$config['sess_expiration'] = 7200;
```

Configuration de la Base de données

```php
<?php

// La valeur TRUE permet d'utiliser la base de données
$config['sess_use_database'] = TRUE;

// Le nom de la table
$config['sess_table_name'] = 'ci_sessions';
```

Le fichier database.php

```php
<?php

/*
| -------------------------------------------------------------------
-
| DATABASE CONNECTIVITY SETTINGS
| -------------------------------------------------------------------
-
*/

$db['default']['hostname'] = "nom_d_hote";
$db['default']['username'] = "nom_d_utilisateur";
$db['default']['password'] = "mot_de_passe";
$db['default']['database'] = "base_de_donnees";
```

Modèle MVC avec CodeIgniter

Nom de la couche	Rôle de la couche
Contrôleurs	C'est le cœur de votre application. Ce sont eux qui seront appelés en premier. Ils seront l'intermédiaire entre les modèles, les vues et toute autre forme de ressources.
Vues	Principalement composées de code HTML, elles permettent de renvoyer le code aux visiteurs.
Modèle	Facultatifs, ils permettent d'envoyer des requêtes à la base de données.

Manipulation de la base de données avec CodeIgniter

La bibliothèque Database

Pour charger la bibliothèque et vous connecter à la base de données, vous devez utiliser la méthode Database du loader.

```php
<?php

$this->load->database();
```

Exemple de requête SQL:

```php
<?php

$resultat = $this->db->select('id, email')
                     ->from('utilisateurs')
                     ->where('pseudo', 'ChuckNorris')
                     ->limit(1)
                     ->get()
                     ->result();
```

Cette requête se lit comme ceci:

« **Sélectionne-moi les colonnes ‘id’ et ‘email’ de la table ‘utilisateurs’ où le champ ‘pseudo’ vaut ‘ChuckNorris’ et arrête-toi dès que tu auras 1 résultat**».

Lorsque nous aurons des requêtes beaucoup plus compliquées, il sera peut-être judicieux de revenir aux requêtes sous forme de chaînes de caractères. Voici un exemple :

Avec la méthode query:

```php
<?php

// Mise en place de notre requête
$sql = "SELECT `id`,
`email`
FROM `utilisateurs`
WHERE `pseudo` = ?
LIMIT 0,1
;";

// Les valeurs seront automatiquement échappées
$data = array('ChuckNorris');

// On lance la requête
$query = $this->db->query($sql, $data);

// On récupère le nombre de résultats
$nb_resultat = $query->num_rows();

// On parcourt l'ensemble des résultats
foreach($query->result() as $ligne)
{
 echo $ligne->id;
}

// On libère la mémoire de la requête (fortement conseillé pour
lancer une seconde requête)
$query->free_result();
```

CONCLUSION

Il était question pour nous de dire de façon basique comment se connecter à une base de données à partir de PHP via PDO et à partir du framework Zend et CodeIgniter. Mais aussi d'y faire des manipulations.

Ceci étant fait ce que nous devons retenir déjà c'est que il n'ya pas de meilleur framework mais lors de l'utilisation d'un framework il faudra penser à la communauté, à l'ancienneté du produit et de ce que disent les personnes sur internet