UNIVERSITE CHEIKH ANTA DIOP DE DAKAR

ECOLE SUPERIEURE POLYTECHNIQUE

DEPARTEMENT INFORMATIQUE

ANNEE UNIVERSITAIRE 2014 | 2015

M. FALL && M. TOURE

DSTI2

Dev_Web_PHP

TD - TP N°8: Bases de données (1)

Exercice 0:

Dans cet exercice, nous allons essayer de créer une application (très simple) dirigée par les données. En premier lieu, elle affiche la liste de tous les membres d'un club de lecture sous forme de tableau html. Le tableau contient des colonnes renseignant le nom d'utilisateur, le prénom et le nom des membre, et vous pourrez trier les données selon chacune des colonnes.
La liste des membres est aussi paginée, affichant seulement 5 membres à la fois ainsi que de liens en dessous de la liste vous permettant de passer à la page suivante ou à la page précédente.
Sur la liste, chaque membre inclut une vue membre sous la forme d'un lien dont un click permet de visualiser toutes les informations sur ce membre, telles que la date d'adhésion, son sexe, son genre favori, son mail, ses centres d'intérêts ainsi que les pages qu'il a eues à visualiser dans le site web du club.
L'application est de type orientée-objet, de ce fait, crée des classes pour gérer les membres et accéder aux bases de données contenant les données de l'application. L'application sera divisée en un ensemble de petits fichiers. Cette approche est celle préconisée par rapport à celle qui utilise un seul fichier pour y stocker l'ensemble des scripts. Cette approche permet entre autres de mieux se localiser et déboguer le code.
Créez d'abord un dossier nommé : ***book_club.***
A l'intérieur, on crée d'abord le fichier nommé : ***config.php.*** Ce fichier contient le code ci-dessous :

```php
<?php
define( "DB_DSN", "mysql:dbname=mydatabase" );
define( "DB_USERNAME", "root" );
define( "DB_PASSWORD", "" );
define( "PAGE_SIZE", 5 );
define( "TBL_MEMBERS", "members" );
define( "TBL_ACCESS_LOG", "accessLog" );
?>
```

Ce fichier définit un ensemble de constantes.

- DB_DSN définit le DSN pour se connecter à la BD MySQL
- DB_USERNAME représente le nom d'utilisateur MySQL à utiliser pour se connecter à la BD
- DB_PASSWORD stocke le ***mdp*** MySQL à utiliser. N'oubliez pas de changer le ***mdp*** et mettre le vôtre.
- PAGE_SIZE spécifie combien de membres seront affichés dans chaque page
- TBL_MEMBERS représente le nom de la table ***member***. Cette façon de faire est très efficace par exemple quand on veut modifier le nom de la table de la BD par exemple.
- TBL_ACCESS_LOG représente le nom de la table ***accessLog.***

Après le fichier, vous pourrez créer le fichier ***common.inc.php.*** Ce fichier ne contient pas grand-chose autre que l'entête et le pied de page XHTML pour toutes pages. Pour l'entête la fonction prend en entrée le titre de la page.

```
1  <?php
2
3  function displayPageHeader( $pageTitle ) {
4  ?>
5  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
6    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
7  <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
8    <head>
9      <title><?php echo $pageTitle?></title>
10     <link rel="stylesheet" type="text/css" href="../common.css" />
11     <style type="text/css">
12       th { text-align: left; background-color: #bbb; }
13       th, td { padding: 0.4em; }
14       tr.alt td { background: #ddd; }
15     </style>
16   </head>
17   <body>
18
19     <h1><?php echo $pageTitle?></h1>
20 <?php
21 }
24 function displayPageFooter() {
25 ?>
26    </body>
27  </html>
28 <?php
29  }
30  ?>
```

Après cela on crée le premier script qui sera réellement manipulé par l'application. On crée le fichier ***DataObject.class.php***. Ce script contient les classes à manipuler dans l'application. Cependant, c'est une classe abstraite, donc pas d'instance de cette classe à créer directement. Les autres classes à créer par exemple, les classes member et dataLog seront créées à partir de cette classe. Le code est donné ci-dessous, on le commente ensemble.

```php
<?php

require_once "config.php";

abstract class DataObject {

  protected $data = array();

  public function __construct( $data ) {
    foreach ( $data as $key => $value ) {
      if ( array_key_exists( $key, $this->data ) ) $this->data[$key] = $value;
    }
  }

  public function getValue( $field ) {
    if ( array_key_exists( $field, $this->data ) ) {
      return $this->data[$field];
    } else {
      die( "Field not found" );
    }
  }

  public function getValueEncoded( $field ) {
    return htmlspecialchars( $this->getValue( $field ) );
  }

  protected function connect() {
    try {
      $conn = new PDO( DB_DSN, DB_USERNAME, DB_PASSWORD );
      $conn->setAttribute( PDO::ATTR_PERSISTENT, true );
      $conn->setAttribute( PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION );
    } catch ( PDOException $e ) {
      die( "Connection failed: " . $e->getMessage() );
    }

    return $conn;
  }

  protected function disconnect( $conn ) {
    $conn = "";
  }
}

?>
```