Programmation orientée objet en PHP

La dao

Nous allons renommer notre classe MysqlDao en PersonneDao puisque le rôle de cette classe est de gérer la persistance des Personne(s).
Ensuite nous allons créer une interface IPersonneDao, qui déclarera les méthodes qui doivent être implémentées par une classe de service dont le rôle sera la persistance des Personne(s) :

```php
<?php

namespace dao;

use entity\Personne;

interface IPersonneDao {

    public function getAllPersonnes();

    public function addPersonne(Personne $personne);
}
```

Afin d'alléger les responsabilités de la classe PersonneDao, et de lui permettre de travailler avec n'importe quel SGBD, nous créons une classe qui gérera la création des connexions. Cette classe de connexion utilisera les configurations du fichier conf.ini. Nous ajoutons à ce fichier une entrée correspondant au type de driver utilisé :

```
[prod]
driver = mysql
host = localhost
db_name = petite_application
user = root
password = root

[dev]
driver = mysql
host = localhost
db_name = test_petite_application
user = root
password = root

[settings]
env = dev
```

La classe Connection ne possédera qu'une méthode de classe getConnection() qui renverra un objet PDO :

```php
<?php

// dao/Connection.php

namespace dao;

use PDO;

class Connection {

    public static function getConnection() {
```

```
        $conf = parse_ini_file(INI_FILE, true);
        $settings = $conf['settings']['env'];
        $host = $conf[$settings]['host'];
        $db_name = $conf[$settings]['db_name'];
        $db_user = $conf[$settings]['user'];
        $db_password = $conf[$settings]['password'];
        $db_driver = $conf[$settings]['driver'];
        $db_datasource = null;
        switch ($db_driver) {
            case 'mysql':
                $db_datasource = "mysql:host=$host;dbname=$db_name";
                break;
        }
        $conn = new PDO($db_datasource, $db_user, $db_password
                , [PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION]);

        return $conn;
    }

}
```

La classe PersonneDao utilisera donc cette classe Connection, et implémentera l'interface IPersonneDao :

```
<?php

// dao/PersonneDao.php

namespace dao;

use PDO;
use dao\IPersonneDao;
use entity\Personne;
use dao\Connection;

class PersonneDao implements IPersonneDao {

    private $conn;

    public function __construct() {
        $this->conn = Connection::getConnection();
    }

    public function getAllPersonnes() {
        $statement = 'SELECT * FROM personne';
        $stmt = $this->conn->prepare($statement);
        $stmt->execute();
        $result = [];
        while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
            $pers = new Personne($row['nom'], $row['prenom'], $row['id']);
            $result[] = $pers;
        }
        return $result;
    }

    public function addPersonne(Personne $personne) {
        $statement = 'INSERT INTO personne (nom,prenom) values '
                . '(:nom, :prenom)';
        $stmt = $this->conn->prepare($statement);
        $stmt->bindValue(':nom', $personne->getNom());
        $stmt->bindValue(':prenom', $personne->getPrenom());
        $result = $stmt->execute();
        return $result;
    }

}
```

Dans le répertoire test/dao nous créons une classe de test ConnectionTest :

```php
<?php

// test/dao/ConnectionTest

include_once './setup.php';
include './dao/Connection.php';

use dao\Connection;

class ConnectionTest extends PHPUnit_Framework_TestCase {

    public function testGetConnection() {
        $conn = Connection::getConnection();
        $this->assertInstanceOf('PDO', $conn);
    }

}
```

Et une classe PersonneDaoTest utilisant les mêmes datastets que le TP précédent :

```php
<?php

// test/dao/PersonneDaoTest.php

include_once './setup.php';
include_once './entity/Personne.php';
include_once './dao/PersonneDao.php';

use entity\Personne;
use dao\PersonneDao;

class PersonneDaoTest extends PHPUnit_Extensions_Database_TestCase {

    protected $connection;
    protected $dao;

    protected function getConnection() {
        if ($this->connection === null) {
            $conf = parse_ini_file(INI_FILE, true);
            $settings = $conf['settings']['env'];
            $host = $conf[$settings]['host'];
            $db_name = $conf[$settings]['db_name'];
            $db_user = $conf[$settings]['user'];
            $db_password = $conf[$settings]['password'];
            $db_driver = $conf[$settings]['driver'];
            $connectionString = "$db_driver:host=$host;dbname=$db_name";
            $this->connection = $this->createDefaultDBConnection(
                    new PDO($connectionString, $db_user
                    , $db_password));
        }
        return $this->connection;
    }

    protected function getDataSet() {
        return new PHPUnit_Extensions_Database_DataSet_YamlDataSet(
                './test/dataset/personne_base.yml');
    }

    protected function setUp() {
        $conn = $this->getConnection();
        // désactivation des contraintes de clés étrangères pour permettre
        //le chargement des données via le dataset
        $conn->getConnection()->query('set foreign_key_checks=0');
        parent::setUp();
```

```
45          // activation des contraintes de clés étrangères
46          $conn->getConnection()->query('set foreign_key_checks=1');
47          $this->dao = new PersonneDao();
48      }
49
50      public function testAddPersonne() {
51          $pers = new Personne("Kent", "Clark");
52          $this->dao->addPersonne($pers);
53          // création d'un objet dataset à partir de la connexion
54          $actualDataset = new PHPUnit_Extensions_Database_DataSet_QueryDataSet(
55                  $this->getConnection());
56          // ajout à ce dataset des enregistrements de la table personne
57          // de notre base de données de test
58          $actualDataset->addTable('personne');
59          // récupération d'un dataset à partir de notre fichier
60          $expectedDataset = new PHPUnit_Extensions_Database_DataSet_YamlDataSet(
61                  './test/dataset/add_personne.yml');
62          // comparaison des deux datasets
63          $this->assertDataSetsEqual($expectedDataset, $actualDataset);
64      }
65
66  }
```

Fin

NAVIGATION

Accueil
- Ma page
  - Pages du site
  - Mon profil
  - Cours actuel
    - POO PHP
      - Participants
      - Généralités
      - La programmation orientée objet : premiers pas
      - L'héritage
      - Les interfaces
      - Le typage
      - Les namespaces
      - Les exceptions
      - Les bases de données avec PDO
      - Les tests avec PHPUnit
      - Petite application version 2
      - Petitie application version 3
      - Le ServiceLocator
      - **La dao**
      - Les classes de traitement métier
      - Les contrôleurs
  - Mes cours

ADMINISTRATION

Administration du cours

Réglages de mon profil

Connecté sous le nom « Arnaud Lemais » (Déconnexion)
POO PHP