

MISE EN PRATIQUE DE LA POO EN PHP 6/6

Auteur : Sébastien Inion

<https://replit.com/@Sebastien11/PersistenceObjets>

AU PROGRAMME DE CE COURS...

Dans cette dernière partie, nous verrons :

- ✱ Comment rendre **persistants** les objets (BDD, fichiers)
- ✱ L'utilisation de la méthode **hydrate()**
- ✱ Faire du CRUD

ETUDE D'UN CAS CONCRET

	NumP	NameP	Address	Salary	
▶	1	Adémar Monoto	3 rue Alex Pyration 31000 Toulouse	3200.00	
	2	Claire Hyère	12 boulevard Gaspard Alizan 1100...	2812.00	
	3	Matéo Porte	12 avenue Yvon Lavallée 34000 M...	4200.00	
	4	Tex Agère	34 rue Milène Micoton 75000 Paris	2770.70	

- ▶ On veut pouvoir créer cette table dans une BDD en SQLite (par exemple)

INSTALLATION DE L'ENVIRONNEMENT

- ▶ **BDD en SQLite** : On va créer une table sur Replit en SQLite pour faire nos tests.
- ▶ Créer la BDD en créant un fichier avec l'extension .sqlite
- ▶ Créer un petit script afin de créer notre table

```
<?php
// Chemin vers votre base de données SQLite
$db_path = __DIR__ . '/bdd/database.sqlite'; // __DIR__ est le répertoire courant du
script PHP

try {
    // Connexion à la base de données SQLite avec PDO
    $pdo = new PDO('sqlite:' . $db_path);

    // Configuration des attributs PDO
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    // Requête SQL pour créer la table PILOT
    $query = "CREATE TABLE IF NOT EXISTS PILOT (
        NumP INTEGER PRIMARY KEY,
        NameP TEXT NOT NULL,
        Address TEXT NOT NULL,
        Salary REAL NOT NULL
    )";

    // Exécution de la requête SQL
    $pdo->exec($query);

    echo "La table PILOT a été créée avec succès.<br>";
} catch (PDOException $e) {
    // Gérer les erreurs éventuelles
    echo "Erreur : " . $e->getMessage();
}

?>
```

INSTALLATION DE L'ENVIRONNEMENT

► Ajouter des données dans notre table

```
<?php
// Chemin vers votre base de données SQLite
$db_path = __DIR__ . '/bdd/database.sqlite'; // __DIR__ est le répertoire courant du script PHP

try {
    // Connexion à la base de données SQLite avec PDO
    $pdo = new PDO('sqlite:' . $db_path);

    // Configuration des attributs PDO
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    // Tableau des données à insérer
    $data = [
        [1, 'Adémar Monoto', '3 rue Alex Pyration 31000 Toulouse', 3200.00],
        [2, 'Claire Hyère', '12 boulevard Gaspard Alizan 11000 Carcassonne', 2812.00],
        [3, 'Matéo Porte', '12 avenue Yvon Lavallée 34000 Montpellier', 4200.00],
        [4, 'Tex Agère', '34 rue Milène Micoton 75000 Paris', 2770.70],
    ];

    // Préparation de la requête d'insertion
    $stmt = $pdo->prepare('INSERT INTO PILOT (NumP, NameP, Address, Salary) VALUES (:NumP, :NameP, :Address, :Salary)');

    // Insertion des données
    foreach ($data as $row) {
        $stmt->bindParam(':NumP', $row[0], PDO::PARAM_INT);
        $stmt->bindParam(':NameP', $row[1], PDO::PARAM_STR);
        $stmt->bindParam(':Address', $row[2], PDO::PARAM_STR);
        $stmt->bindParam(':Salary', $row[3], PDO::PARAM_STR);
        $stmt->execute();
    }

    echo "Les données ont été insérées avec succès.";
} catch (PDOException $e) {
    // Gérer les erreurs éventuelles
    echo "Erreur : " . $e->getMessage();
}
?>
```

MISE EN APPLICATION...

PASSER DE LA BDD À L'OBJET ET DE L'OBJET À LA BDD

- ▶ Finalement une ligne dans ma BDD correspond à un objet !
- ▶ Chaque champs est un attribut de mon objet.
- ▶ On va donc créer une classe **Pilote** en se basant sur la structure de la table en BDD.
- ▶ Pour les méthodes on fera comme d'habitude les *getters* et *setters* nécessaires.
- ▶ **ATTENTION !!!** On ne doit trouver aucune trace de persistance d'objet ici (BDD, fichier JSON, etc)
- ▶ **En POO une classe un rôle !** La classe Pilote doit permettre de créer les objets en mémoire vive.
- ▶ Pour ce qui est de la persistance on fera une autre classe **PiloteManager**

LA CLASSE PILOTE

- ▶ La classe Pilote est classique !
 - ▶ Ce qui change est l'introduction de la méthode hydrate !
- ▶ Son rôle créer des objets pilotes !

```
<?php
class Pilote {
    private $NumP;
    private $NameP;
    private $Address;
    private $Salary;

    public function __construct(array $data = []) {
        if (!empty($data)) {
            $this->hydrate($data);
        }
    }

    public function hydrate(array $data) {
        foreach ($data as $key => $value) {
            $method = 'set' . ucfirst($key);
            if (method_exists($this, $method)) {
                $this->$method($value);
            }
        }
    }

    public function getNumP() {
        return $this->NumP;
    }

    public function getNameP() {
        return $this->NameP;
    }

    public function getAddress() {
        return $this->Address;
    }

    public function getSalary() {
        return $this->Salary;
    }
}
```


L'hydratation

```
public function hydrate(array $data) {  
    foreach ($data as $key => $value) {  
        $method = 'set' . ucfirst($key);  
        if (method_exists($this, $method)) {  
            $this->$method($value);  
        }  
    }  
}
```

- **L'hydratation** c'est le fait d'hydrater d'objet c'est à dire de lui apporter ce qu'il faut pour qu'il fonctionne ici les données pour les propriétés.
- Dans ce cas le constructeur se contente d'appeler la méthode hydrate (qui n'est pas une méthode magique !)
- Comme les attributs portent le nom des champs dans la base de données on va appeler les setters au fur et à mesure (boucle)
- Cette méthode prend en paramètre un tableau qui contient toutes les données.
- Remarque : hydrate() passe par les setters

LA CLASSE PILOTEMANAGER

```
<?php
class PiloteManager {
    private $pdo;

    public function __construct($db_path) {
        try {
            $this->pdo = new PDO('sqlite:' . $db_path);
            $this->pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        } catch (PDOException $e) {
            echo "Erreur : " . $e->getMessage();
        }
    }

    public function addPilote(Pilote $pilote) {
        try {
            $query = 'INSERT INTO PILOT (NumP, NameP, Address, Salary) VALUES (:NumP, :NameP, :Address, :Salary)';
            $stmt = $this->pdo->prepare($query);
            $stmt->bindValue(':NumP', $pilote->getNumP(), PDO::PARAM_INT);
            $stmt->bindValue(':NameP', $pilote->getNameP(), PDO::PARAM_STR);
            $stmt->bindValue(':Address', $pilote->getAddress(), PDO::PARAM_STR);
            $stmt->bindValue(':Salary', $pilote->getSalary(), PDO::PARAM_STR);
            $stmt->execute();
        } catch (PDOException $e) {
            echo "Erreur : " . $e->getMessage();
        }
    }
}
```

- ▶ La classe PiloteManager
- ▶ Son rôle faire le lien avec la BDD
- ▶ On y trouvera toutes les méthodes nécessaires pour travailler et faire du mapping
 - ▶ mapping :
Objet -> ligne dans la BDD
ligne dans la BDD -> Objet
- ▶ Ici on parle de BDD mais on pourrait travailler avec par exemple un fichier JSON !

UTILISATION DU CRUD...

ON PEUT MAINTENANT TESTER

- ▶ Et faire du CRUD par exemple (Create Read Update Delete)

CREATE

```
public function addPilote(Pilote $pilote) {
    try {
        $query = 'INSERT INTO PILOT (NumP, NameP, Address, Salary) VALUES (:NumP, :NameP, :Address, :Salary)';
        $stmt = $this->pdo->prepare($query);
        $stmt->bindValue(':NumP', $pilote->getNumP(), PDO::PARAM_INT);
        $stmt->bindValue(':NameP', $pilote->getNameP(), PDO::PARAM_STR);
        $stmt->bindValue(':Address', $pilote->getAddress(), PDO::PARAM_STR);
        $stmt->bindValue(':Salary', $pilote->getSalary(), PDO::PARAM_STR);
        $stmt->execute();
    } catch (PDOException $e) {
        echo "Erreur : " . $e->getMessage();
    }
}
```

ON PEUT MAINTENANT TESTER

- ▶ Et faire du CRUD par exemple (Create Read Update Delete)

READ

```
public function getPilote($NumP) {  
    try {  
        $query = 'SELECT * FROM PILOT WHERE NumP = :NumP';  
        $stmt = $this->pdo->prepare($query);  
        $stmt->bindValue(':NumP', $NumP, PDO::PARAM_INT);  
        $stmt->execute();  
        $data = $stmt->fetch(PDO::FETCH_ASSOC);  
        if ($data) {  
            return new Pilote($data);  
        }  
        return null;  
    } catch (PDOException $e) {  
        echo "Erreur : " . $e->getMessage();  
    }  
}
```

ON PEUT MAINTENANT TESTER

- ▶ Et faire du CRUD par exemple (Create Read Update Delete)

UPDATE

```
public function updatePilote(Pilote $pilote) {
    try {
        $query = 'UPDATE PILOT SET NameP = :NameP, Address = :Address, Salary = :Salary WHERE NumP = :NumP';
        $stmt = $this->pdo->prepare($query);
        $stmt->bindValue(':NumP', $pilote->getNumP(), PDO::PARAM_INT);
        $stmt->bindValue(':NameP', $pilote->getNameP(), PDO::PARAM_STR);
        $stmt->bindValue(':Address', $pilote->getAddress(), PDO::PARAM_STR);
        $stmt->bindValue(':Salary', $pilote->getSalary(), PDO::PARAM_STR);
        $stmt->execute();
    } catch (PDOException $e) {
        echo "Erreur : " . $e->getMessage();
    }
}
```

ON PEUT MAINTENANT TESTER

- ▶ Et faire du CRUD par exemple (Create Read Update Delete)

DELETE

```
public function deletePiloteNum($NumP) {
    try {
        $query = 'DELETE FROM PILOT WHERE NumP = :NumP';
        $stmt = $this->pdo->prepare($query);
        $stmt->bindValue(':NumP', $NumP, PDO::PARAM_INT);
        $stmt->execute();
    } catch (PDOException $e) {
        echo "Erreur : " . $e->getMessage();
    }
}

public function deletePilote(Pilote $pilote) {
    $pilote->getNumP();
    $this->deletePiloteNum($pilote->getNumP());
}
```




**MERCI POUR
VOTRE ATTENTION**