

## TP4 – Premiers pas en programmation objet en PHP, Premier formulaire avec traitement par PHP

### Conventions internes sur le nommage :

- Nous nommerons nos fichiers PHP en minuscules. Par exemple, auteur.php ou bd.php.
- En PHP, le nom d'une classe n'est pas sensible à la casse (minuscule/majuscule). Nous pourrions partir sur un nommage des classes en minuscules. Par exemple, la classe **bd** qu'on déclarera dans le fichier bd.php.
- Quand nous avons développé nos tables de données sur le serveur de données (via l'interface phpMyadmin), nous avons nommé nos tables en minuscules : la table **bd** par exemple. Attention, le nommage des tables de données est sensible à la casse. Il faudra y faire attention dans l'écriture des futures requêtes.

### Contexte et objectifs :

- Dans ce TP, au début, nous créons les classes PHP en cohérence avec le travail réalisé au TP3.
- Ensuite, nous fabriquons des instances des différentes classes PHP, sans lien avec la base de données (l'utilisation des différentes données de la base sera l'objet du TP5).
- Nous parlerons d'héritage de classes.

## Exercice 1 – La classe `adherent` et la page de test

1. Créer dans un fichier `TP4/ex1/adherent.php` une classe **`adherent`** avec :

- Le constructeur qui déclare les 6 attributs (`protected`) d'instance `$login`, `$mdp`, `$nomAdherent`, `$prenomAdherent`, `$email` et `$telephone`.
- Le `getter`, et le `setter`. Vous utiliserez une version générique comme dans le cours.
- La méthode `__toString()` qui retourne une chaîne de caractères donnant quelques caractéristiques de l'adhérent, sous la forme

**adhérent speedy (Antoine Dupont)**

chaîne de caractères composée du login, du prénom et du nom de l'adhérent)

Voir le cours pour un exemple de code classique :

2. Créer un fichier `TP4/ex1/testAdherent.php` qui permettra de créer plusieurs adhérents par un appel au constructeur, et qui affichera ces adhérents. Donnez une structure html à ce fichier et insérez votre partie script PHP dans le body. Voici le rendu attendu :



N'oubliez pas d'insérer la classe **`adherent`** pour commencer, sinon l'appel au constructeur provoquera une erreur.

## Exercice 2 – La classe *categorie* et la page de test

Refaites exactement pareil avec les catégories :

- Un fichier `categorie.php` qui définit la classe ***categorie***, comme pour la classe *adherent*,
- Un fichier de test qui crée et affiche quelques instances de la classe ***categorie*** :



## Exercice 3 – Une classe mère pour coder les méthodes communes

Les classes ***adherent*** et ***categorie*** ont en commun des méthodes que nous allons faire migrer vers une classe mère, que nous allons appeler ***objet***, et qui contiendra pour le moment les getter et setter.

```
<?php
class objet {
    public function get($attribut) {
        return $this->$attribut;
    }

    public function set($attribut, $valeur) : void {
        $this->$attribut = $valeur;
    }
}
?>
```

1. Codez cette classe **objet** dans un fichier `ex3/objet.php`. Vous remarquerez que `$attribut`, chaîne de caractères, va remplacer tous les attributs possibles dans le `return`. Autre chose : on peut se dispenser du type de retour pour le `getter`. Ce type est en fait variable.
2. Faites hériter les classes **adherent** et **categorie** de la classe **objet**, et supprimant des classes filles les `getter` et `setter` particuliers devenus inutiles. Voici un exemple de code pour **categorie** :

```
<?php
require_once("objet.php");
class categorie extends objet {

    protected int $numCategorie;
    protected string $nomCategorie;
    protected string $descriptif;

    public function __construct(int $numCategorie, string $nomCategorie, string $descriptif) {
        $this->numCategorie = $numCategorie;
        $this->nomCategorie = $nomCategorie;
        $this->descriptif = $descriptif;
    }

    public function __toString() : string {
        $n = $this->nomCategorie;
        $d = $this->descriptif;
        return "catégorie $n ($d)";
    }

}
?>
```

Vous remarquerez deux choses importantes :

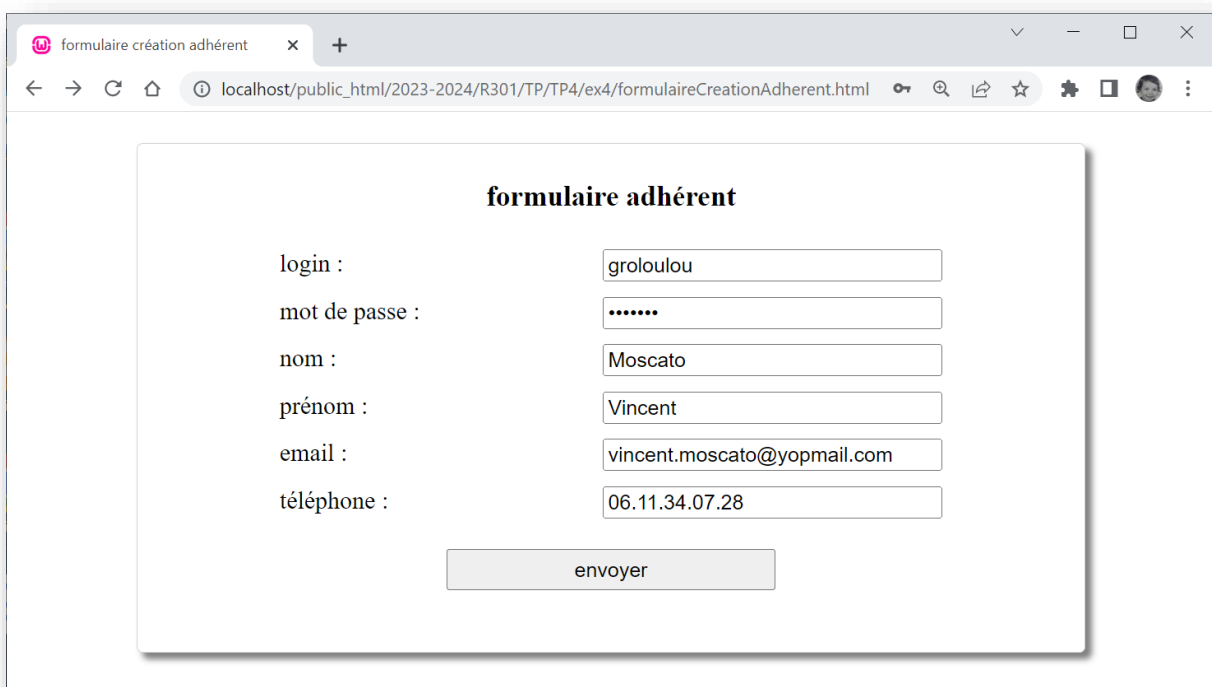
- On inclut le fichier contenant la classe **objet**, pour que l'expression « `extends objet` » ait un sens,
  - Dans le constructeur, on choisit une visibilité « `protected` », comme c'est le cas en java par exemple avec les héritages. Si vous laissez les visibilités « `private` », vous aurez des erreurs d'accessibilité relevées par PHP.
3. Maintenant que vos deux classes héritent de la classe **objet**, recopiez le fichier `ex1/testAdherent.php` dans `ex3/testAdherent.php` de façon à obtenir le même résultat que précédemment.

## Exercice 4 – Le premier formulaire, pour créer un adhérent

Nous allons mettre en place un formulaire html qui permettra de saisir les éléments qui constituent un adhérent. Ce formulaire sera lié à un fichier de traitement, qui recevra les données, créera le nouvel adhérent, et l'affichera.

Le code html du formulaire vous est proposé sur Moodle, ainsi qu'un fichier css. Vous êtes libre de personnaliser le css, mais le formulaire est conçu de façon standard, il est donc conseillé de le garder sous sa forme de base.

1. Dans votre dossier ex4, recopiez les fichiers `objet.php`, `adherent.php`, `categorie.php`. Copiez aussi les fichiers `formulaireCreationAdherent.html` et `styles.css`.
2. Après avoir comme d'habitude transféré vos fichiers sur le serveur, ouvrez le formulaire, complétez-le et cliquez sur envoyer.

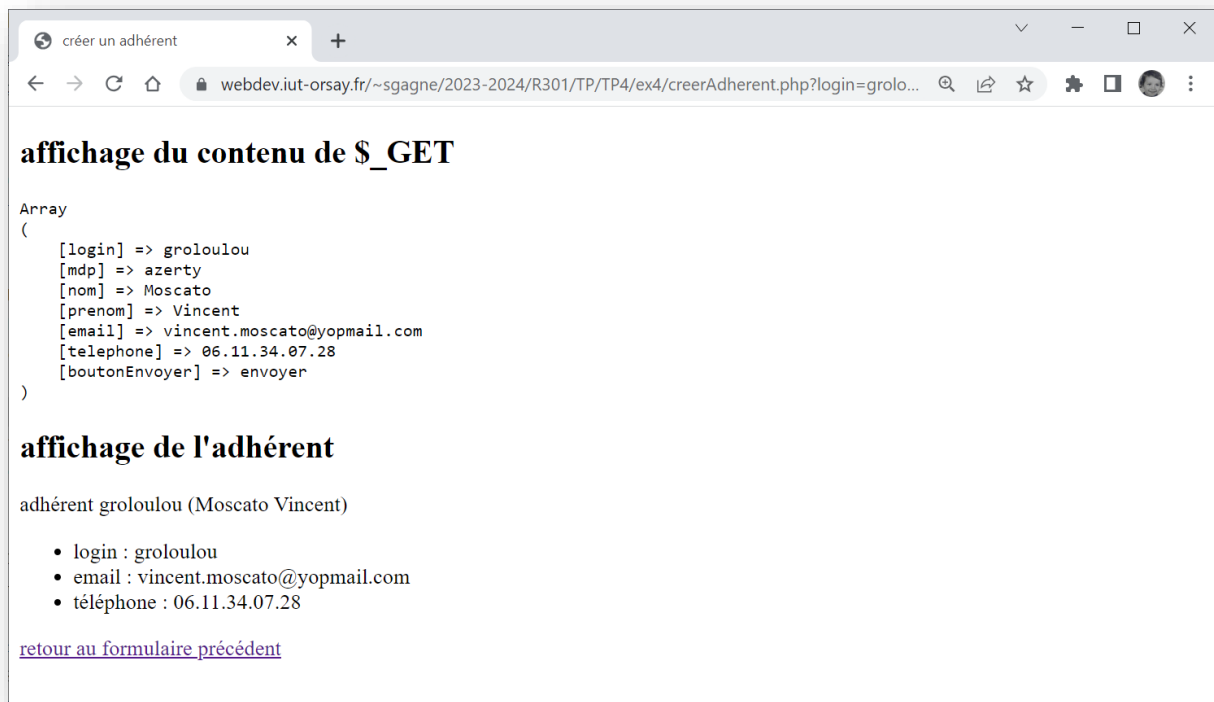


The screenshot shows a web browser window with the title 'formulaire création adhérent'. The address bar shows the URL 'localhost/public\_html/2023-2024/R301/TP/TP4/ex4/formulaireCreationAdherent.html'. The form itself is titled 'formulaire adhérent' and contains the following fields:

Label	Value
login :	groloulou
mot de passe :	.....
nom :	Moscato
prénom :	Vincent
email :	vincent.moscato@yopmail.com
téléphone :	06.11.34.07.28

At the bottom of the form is a button labeled 'envoyer'.

3. Expliquer l'erreur produite en examinant l'url de la page de traitement.
4. Pour corriger l'erreur, créez le fichier manquant, comme c'est expliqué dans le cours. Le résultat attendu est donné dans la capture suivante. Vous récupérerez les éléments de `$_GET`, puis vous construirez un nouvel adhérent (avec le constructeur de la classe ***adherent***), que vous afficherez, avec quelques détails.



Vous remarquerez le lien en bas de page qui permet de revenir au formulaire.

## Exercice 5 – Les autres classes

Codez les autres classes : ***auteur***, ***bd***, ***etat***, ***serie***, ***nationalite***, ***est\_auteur\_de***, chaque classe dans un fichier spécifique.

Vous ferez hériter les classes ***auteur***, ***bd***, ***etat***, ***serie***, ***nationalite*** de la classe ***objet***.

Pour la classe ***est\_auteur\_de***, pas besoin d'héritage, c'est une classe à part.