



# Programmation orientée objet en PHP

[Accueil](#) ► [Mes cours](#) ► [Développement logiciel](#) ► [POO PHP](#) ► [Petite application version 2](#) ► [Le projet](#)

## Le projet

Dans ce TP nous allons reprendre une architecture en couche comme dans le module précédent. Nous retrouverons donc les couches dao, controller et view.

Nous allons ajouter à cette architecture un front controller, qui servira de point d'entrée unique à l'application. Lors la réception d'une requête HTTP, la requête sera redirigée vers [le front controller](#), puis [le front controller](#) sollicitera à son tour un controller adéquate.

L'un des intérêts de cette architecture est qu'elle permet une conception orientée objet de l'application. Les controllers pourront donc être implémentés sous forme de classes.

Des classes utilitaires pourront venir compléter cette architecture, telles que des classes "routers" gérant les "routes" (c'est-à-dire les correspondances entre les URL(s) et [les contrôleurs](#)), des classes gérant les formulaires et leurs validations, etc...

Pour ce projet nous allons utiliser le serveur web interne de PHP. Pour démarrer notre serveur, dans un terminal nous nous plaçons dans le répertoire web de notre projet, puis nous exécuterons la commande `php -S localhost:8080 router.php`. En passant en paramètre le fichier router.php contenu dans le répertoire web, nous demanderons à notre serveur interne de passer tout d'abord par ce script quelque soit l'URL demandée. Si l'URL demandée correspond à une ressource (image, css...) alors le script router.php renverra null et l'URL sera traitée normalement. Si la ressource ne correspond pas à une ressource alors [le front controller](#) entrera en action. Le script router.php jouera ici le rôle du fichier .htaccess du serveur HTTP Apache.

```
1 <?php
2
3 // web/router.php
4 if (preg_match('/\.(png|jpg|jpeg|gif|css)$/ ', $_SERVER["REQUEST_URI"])) {
5     return false;
6 } else {
7     include 'index.php';
8 }
```

Le projet sera constitué de trois vues :

- une vue "accueil" avec deux liens => <http://localhost:8080/>
- une vue de consultation des personnes => <http://localhost:8080/personnes>
- une vue d'ajout d'une personne => <http://localhost:8080/personnes/ajouter>
- une vue d'erreur

L'accueil

## Accueil !

[Consulter les personnes](#)  
[Ajouter une personne](#)

La consultation des personnes :

	Id	Nom	Prénom
1	Sparrow	Jack	
2	Wayne	Bruce	
3	Parker	Peter	

4 Kirk James T.  
[Ajouter une personne](#)

L'ajout d'une personne :

Nom :   
 Prénom :

La page d'erreur :

Oups...

Nous allons créer un projet nommé `petite_application`. Dans ce projet nous allons créer un répertoire pour chaque couche applicative (dao, controller, view), un répertoire config qui contiendra le fichier de configuration ainsi que le fichier de routing, un répertoire entity qui contiendra les entités du modèle, un répertoire web qui contiendra toutes les ressources publiques de notre application (images, css...) mais aussi [le front controller](#) (index.php), un répertoire util qui contiendra les classes utilitaires de routing, et enfin le répertoire test qui contiendra tous les tests de l'application. Nous aurons besoin de bibliothèques de test (PHPUnit, DBUnit et PHPUnit-selenium), que nous installerons via composer. L'archive `composer.phar` sera à la racine de notre application ainsi que le fichier `composer.json`.

```

1 {
2     "require-dev": {
3         "phpunit/phpunit": "4.*",
4         "phpunit/dbunit": "1.*",
5         "phpunit/phpunit-selenium": "1.*"
6     }
7 }
```

Une fois les dépendances installées via composer, l'arborescence du projet sera la suivante :

```

petite_application
├── Source Files
│   ├── config
│   ├── controller
│   ├── dao
│   ├── entity
│   ├── test
│   ├── util
│   ├── vendor
│   ├── view
│   ├── web
│   ├── composer.json
│   ├── composer.lock
│   ├── log.txt
│   ├── setup.php
│   └── Include Path
```

Nous créerons pour ce TP deux bases de données MySQL : une base de production nommée `petite_application` et une base de test nommée `test_petite_application`.

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

 **Le projet**

 conf.ini et setup.php

 Les entités et la dao

 Test de la DAO

 La gestion des routes

 Test des routes

 Les contrôleurs

 Les vues

 Le front controller

 Test de l'ajout d'une personne

 T.P. mettre à jour une personne

Petite application version 3

[Mes cours](#)

## ADMINISTRATION



Administration du cours

Réglages de mon profil

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