# Structurer une application PHP en MVC

## Définition MVC

Le MVC est un design pattern (patron de conception) qui consiste à découper votre application en 3 parties distincte :

M => Model : c'est la partie qui s'occupe de l'interaction avec la base de données.

V => Vue : c'est la partie qui s'occupe de l'affichage.

C => Controller : c'est le cœur de l'application, c'est lui qui calcule et gère les différentes données à donner à la ou les vues. Il peut y avoir plusieurs Controllers !!

## La méthode !!

Nommer, ranger, retrouver !

Nommer correctement les fichiers, dans cette structure je nomme avec c_ devant un Controller, v_ devant une vue et m_ devant un model.

Ne pas mettre n'importe quel fichier n' importe où !

/!\ PAS DE TRAITEMENT DANS LES VUES !!!!

## Comment faire en PHP le plus simplement possible ?

### Mettre en place des Controllers

On va tout simplement définir un point d'entrée et gérer les cas à partir de celui-ci. Notre point d'entrée sera l'index.php.

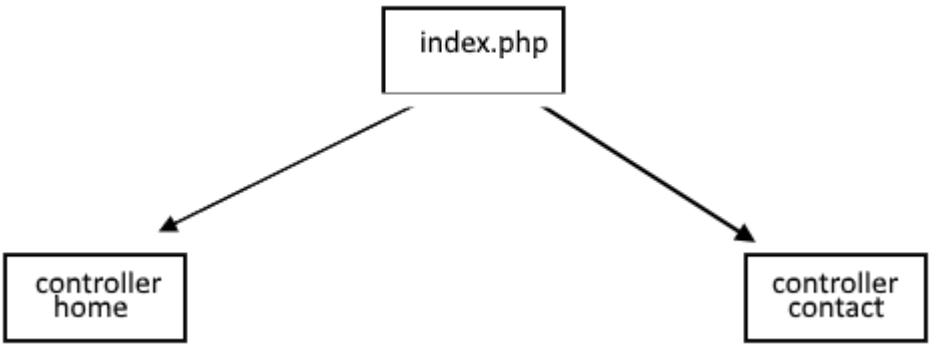
A partir de celui on redirigera sur le bon Controller.

Comment savoir quel Controller le client veut ?

On va utiliser l'URL, les paramètres passés en URL peuvent être récupérer avec la variable $_GET de PHP.

Exemple : http://localhost/?c=home

Petit schéma :



Mais comment choisir le bon Controller ?

Tout simplement en utilisant un SWITH :

```php
!isset($_REQUEST['c']) ? $_controler = 'home' : $_controler = $_REQUEST['c'];

switch ( $_controler ) {
    case 'login' :
        include "c_login.php";
        break;
    default :
        include "c_home.php";
        break;
}
```

Ici à l'arrivé dans l'index.php on regarde :

Mais tu n'utilises pas $_GET ! (oui ce n'est pas dans le cours mais sur php.net), le résultat sera le même.

$_REQUEST => Un tableau associatif qui contient par défaut le contenu des variables $_GET, $_POST et $_COOKIE.

1. J'ai un Controller dans l'URL ? Si oui je le stocke sinon je mets à une valeur Controller par défaut
2. Je switch sur cette variable contenant le Controller
3. Je charge le bon fichier PHP

C'est bien mais un Controller pour une page ça va faire beaucoup de Controller !

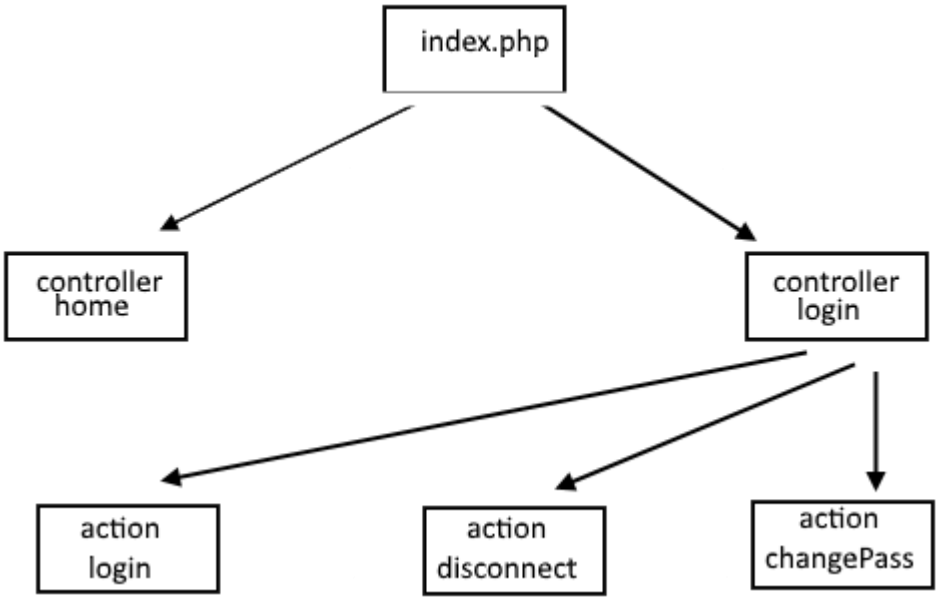
## Les actions c'est quoi ?

Les actions c'est une sous partie d'un Controller.

Par exemple pour un Controller login, ça peut être connexion, déconnexion, ou encore changer de mot de passe.

Comment gérer ça ? Encore une fois avec l'URL et un switch mais dans le Controller cette fois !

Exemple : http://localhost/?c=home&a=login ou http://localhost/?c=home&a=changePass




```php
if(!isset($_REQUEST['a'])) $action = 'null'; else $action = $_REQUEST['a'];

switch($action) {
    case 'disconnect':
        // deconnexion, suppression de la session
        // rediriger sur la page de login
        break;
    case 'postLogin': // URL ou le formulaire de connexion sera post
        // traitement pour la connexion
        // mettre en session
        // rediriger sur le panel admin
        break;
    case 'changePass':
        include '/login/v_changePass.php';
        break;
    case 'postChangePass':
        // traitement changePass
        // mettre a jour en session
        // rediriger sur le panel admin
        break;
    default: // action login
        include '/login/v_connexion.php';
        break;
}
```

Ici par défaut la page de login sera affiché !

Ici à l'arrivé dans le c_login.php on regarde :

1. J'ai une action dans l'URL ? Si oui je le stocke sinon je mets à une valeur action par défaut
2. Je switch sur cette variable
3. Je charge le bon fichier PHP ou on fait les traitements appropriée

## Les models

Les models sont toutes les fonctions utilisées pour faire des interactions avec la base de données.

Les fonctions seront appelées dans les Controllers principalement.

```php
$db = new PDO(DB_HOST.';'.DB_NAME, DB_LOGIN, DB_PASS);
$db->query("SET CHARACTER SET utf8");

// fonction de connexion
function connexion($login,$pass){
    global $db;
    //netoyage des valeurs pour eviter les failles SQL !
    $login = $db->quote($login);
    $pass = $db->quote($pass);
    $pass = md5($pass);
    $q_log="SELECT count(*) as nb , `id` FROM `login` WHERE `login`=".$login." AND `pass`='".$pass."' ;";
    $q_res = $db->query($q_log);
    $a_log = $q_res->fetch(PDO::FETCH_ASSOC);
    return $a_log;
}
```

La partie connexion doit se faire dans le fichier index.php pour être globale à toute l'application.

Ici par exemple la fonction connexion peut être utilisée dans le Controller login dans l'action postLogin.

## Comment tout mettre ensemble

Voici un exemple de structure que j'utilise pour le projet :

Pour mieux voir l'architecture rdv sur :

https://github.com/Tilican/mvcSimple