PHP

J. ROMAGNY

Table des matières

1. Documentations PHP

- Documentation PHP
- The right way
- W3School

2. Les prérequis

a. Installation

Documentation

En **local**, on peut aussi utiliser un « **bundle** » :

- Wamp sur Windows
- Mamp sur Mac
- Lamp sur Linux

Il existe également XAMPP (Mac, Windows, Linux)

Configuration Apache

Exemple changer le port par défaut (80) … pour un autre port (8080 par exemple)

httpd.conf (dossier conf d'Apache)

```
#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.
#
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.
#
#Listen 12.34.56.78:80
Listen 0.0.0.0:8080
Listen [::0]:8080
```

C'est également dans ce fichier que l'on peut activer les modules : exemple décommenter

« LoadModule rewrite_module modules/mod_rewrite.so » pour activer le module « rewrite_mod » et relancer les services.

b. EDI

- PHPStorm (payant, version d'essai 30 jours) offre un environnement de développement robuste pour PHP.
- **Netbeans** (gratuit)
- Etc.

On peut également utiliser des éditeurs de texte comme Brackets ou Sublime Text, pratiques pour ouvrir rapidement un fichier par exemple.

3. COMPOSER

- Composer est un **gestionnaire de packages** pour PHP.
- Packagist permet de chercher des **packages** pour ses projets.

Installation

Sur **Windows** on peut installer Composer avec un exécutable. L'avantage c'est que l'on aura la **commande** « **composer** » disponible en invite de commandes Windows.

Sinon, que ce soit sur **Windows, Mac ou Linux** on peut installer **composer.phar** dans son **projet**. Pour cela copier-coller le contenu de « command line installation ».

Création de « composer.json » du projet

```
composer init
```

Exemple (schema):

```
{
    "name": "romagny13/my-lib",
    "type": "library",
    "description": "My library",
    "keywords": ["my","lib"],
    "license": "MIT",
    "authors": [
        {
            "name": "Jerome Romagny",
            "email": "romagny13@yahoo.fr"
        }
    ],
    "autoload": {
        "psr-4": {
            "App\\": "src"
        }
    },
    "require": {},
    "require-dev": {
        "phpunit/phpunit": "^5.7"
    },
    "scripts":{
        "test": "phpunit -c ."
    }
}
```

Ajout de dépendances

Dans « composer.json »

Directement dans « **composer.json** » dans les sections « **require** » et « **require-dev** »

```
{
    "name": "romag/composer-demo",
    "require": {
        "illuminate/database": "5.4"
    }
}
```

Puis entrer les commandes : « composer install » ou « composer udpate » (ou « php composer.phar install » ou « php composer.phar update »)

Avec les commandes

```
composer require <name>
```

Exemple : « composer require illuminate/database »

Avec une version spécifique : « composer require illuminate/database**:5.0** »

Installation en **global**

```
composer global require <name>
```

Installation en **dépendance de développement**

```
composer require <name> --dev
```

Définir la version minimum de PHP pour le projet

```
"require": {
  "php": ">=5.4.0",
```

Autoloader

Documentation

L'Autoloader permet d'éviter d'avoir à ajouter des require/include au niveau de tous les fichiers.

Dans « composer.json » définir le namespace correspondant au dossier

```
"autoload":{
      "psr-4":{
        "App\\": "src"
      }
  },
```

Il est possible également d'ajouter des fichiers n'ayant pas de classe pour l'Autoloader

```
"autoload":{
      "psr-4":{
        "App\\": "src"
      },
      "files": ["src/functions.php"]
  },
```

Faire « **composer install** » pour **installer l'Autoloader de Composer**

Faire « **composer dump-autoload** » **après une modification de la section « autoload »** de « composer.json »

Au niveau de l'**index.php** ajouter l'autoloader de composer

```
<?php
require __DIR__ . '/../vendor/autoload.php';

session_start();
```

Il est possible de créer son propre Autoloader avec les fonctions « **__autoload** » ou « **spl_autoload_register** » (recommandée)

Exemple dans un fichier nommé « MyAutoloader.php »

```
<?php
function __autoload($name) {
    $file_name = "$name.php";
    if(file_exists($file_name)){
        require_once($file_name);
    }
    else {
        throw new Exception("Cannot load $name");
    }
}
```

Ou

```
<?php
spl_autoload_register(function ($name) {
    $file_name = "$name.php";
    if(file_exists($file_name)){
        require_once($file_name);
    }
    else {
        throw new Exception("Cannot load $name");
    }
});
```

Utilisation (index.php)

```
<?php
require __DIR__ . '/MyAutoloader.php';

use App\MyClass;

$c = new MyClass();
$c->doSomething();
```

Scripts

Dans « composer.json ». Exemple pour lancer un serveur de développement (pointant sur l'index.php + .htaccess dur répertoire « public ») et les tests

```
"scripts": {
        "start": "php -S localhost:8080 -t public",
        "test": "phpunit -c ."
    }
```

Utilisation : « composer start » et « composer test »

4. Packagist

Submit un package

Github

Créer un **répertoire** pour le projet sur Github et créer une **release** (ne pas l'indiquer en pre-release)

Packagist

.. Puis aller sur Packagist … cliquer sur le bouton « **submit** »

.. Entrer l'url du répertoire (exemple « https://github.com/romagny13/my-demo.git ») … choisir le nom sous lequel sera disponible le package.

Le package sera alors installable avec une commande par exemple :

```
composer require romagny13/my-demo
```

5. Virtual Host

Exemple avec **Wamp**

a. C:\wamp\bin\apache\apache2.4.9\conf\extra**httpd-vhosts.conf**

Ajouter

```
<VirtualHost *:80>
   DocumentRoot "C:/wamp/www/rest_simple"
   ServerName local-rest.dev
   ErrorLog "logs/local-rest-error.log"
   CustomLog "logs/local-rest-access.log" common
</VirtualHost>

<VirtualHost *:80>
   DocumentRoot "C:/wamp/www/"
   ServerName localhost
   ServerAlias 127.0.0.1
</VirtualHost>
```

Chemin vers le répertoire du site. Pour un site fait avec Laravel on pourrait indiquer le répertoire « public »

On accèdera au site avec « http://local-rest.dev » dans l'exemple

Note : il est possible de définir dans une balise directory les règles que l'on définirait dans le .htaccess du site.

b. C:\Windows\System32\drivers\etc**hosts**

Ajouter tout en bas sauvegarder le fichier par exemple sur le bureau puis écraser le fichier existant

```
127.0.0.1       local-rest.dev
```

c. C:\wamp\bin\apache\apache2.4.9\conf**httpd.conf**

Dé commenter la ligne (enlever le #)

```
Include conf/extra/httpd-vhosts.conf
```

d. Bases du langage PHP

PHP est un langage faiblement typé

a. Séparation des instructions

```
<?php
 //
?>
```

La balise fermante est optionnelle, il sera parfois bon de ne pas la mettre avec « include », « require » par exemple.

Astuce < ?= équivaut à < ?php echo …

Exemple

```
<?php
$hello = "hello!";
?>
<?= $hello ?>
```

b. Debug

Afficher rapidement le contenu d'une variable

```
var_dump($myvar);
die;
```

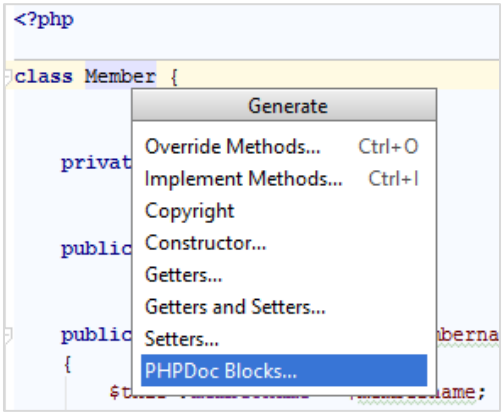
die interrompt l'éxécution

c. Commentaires

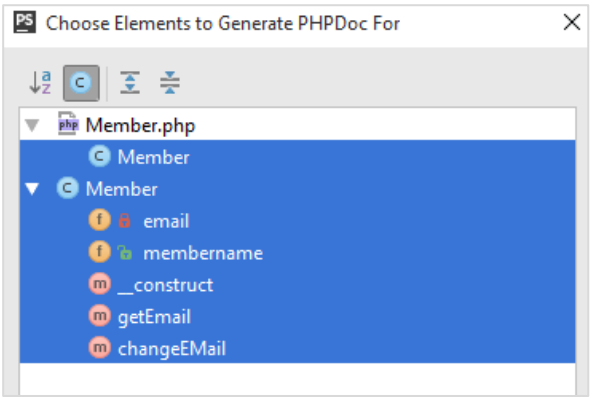
```
// sur une ligne

/*
Sur plusieurs lignes
*/
```

PHPDoc

- Commencer à saisir « /** » au-dessus du membre puis taper sur « entrée »
- Avec **PHPStorm** on peut générer automatiquement …« ALT + INS »



…



d. Variables

```
$mybool = true;
$myint =10;
$myfloat =9.99;
$mystring = "Bonjour!";
$var = null;
```

Savoir si une variable est « **null** » avec is_null()

```
var_dump(is_null($var));
```

Obtenir le **type** d'une variable

```
gettype($mystring)
```

Comparaison

```
if($a==$b){

}
if($a===$b){
}
```

Même valeur

Même valeur et même type

Concaténation de chaines avec « . »

```
echo "Bonjour"." !";
```

Global

```
$myint = 10;

function dosomething(){
 global $myint;
    $myint=100;

}
dosomething();
echo $myint;
```

Sans global $myint serait considérée comme une variable locale à la fonction

$myint vaudra 100

Modification de types

(int) ou (integer), (bool ou (boolean), (float), (double), (real) ,(string) ,(array) ,(object) ,(unset) permet de transtyper une variable vers null

```
$myint =10;
$mystring = (string) $myint;
```

Constantes

```
define("MYCONST", "valeur de ma constante");
echo MYCONST;
```

Tester ses variables

isset($myvar) … vérifie que la variable n'est pas null

is_null($myvar) … vérifie si la variable est null

+ empty$myvar), is_string($myvar), …

e. Constantes

Définition puis utilisation d'une constante :

```
<?php
define("MYCONST", "Valeur de constante");
echo MYCONST;
```

f. Chaines de caractères

Documentation

Concaténation

```php
$person = "Marie";
echo 'Bonjour '.$person;
// ou
echo "Bonjour $person";
```

Trim (+ ltrim et rtrim)

Peut servir à tronquer les espaces ou à retirer un caractère/chaine dans une autre chaine

```php
$url = '/articles/';
$url = trim($url,'/'); // donnerait articles
```

replace

Remplacer un caractère/chaine dans une autre chaine

```php
$result = str_replace('rouge','bleue','La voiture est rouge'); // donne la voiture est bleue
```

explode pour split

```php
$url = trim('/articles/view/10', '/');
$result = explode('/',$url); // donne un tableau avec articles, view et 10
```

On peut faire l'opération inverse avec implode

```php
$tab = ["articles","view","10"];
$result = implode('/',$tab);
```

Expressions régulières

Documentation

g. Conditions

```
if($a==$b && $a>=0){

}
elseif($a<$b){
}
else {
}
```

Utiliser « || » pour « ou »

Autre écriture

```
<?php if($a==$b): ?>
<p>Oui</p>
<?php else: ?>
<p>Non</p>
<?php endif ?>
```

Ternaire

```
$result = ($a==$b) ? true : false;
```

Switch

```
switch ($a)
{
    case 0:
        echo "...";
        break;
    default:
        echo "...";
}
```

h. Boucles

Foreach

```
$fruits = Array("Pomme","Poire","Pêche");
foreach($fruits as $fruit)
{
    echo $fruit . '<br />';
}
```

```
<?php foreach($fruits as $fruit): ?>
   <P>...</P>
<?php endforeach ?>
```

For

```
for ($i = 1; $i <= 10; $i++)
{
    echo "i vaut $i<br/>";
}
```

```
<?php for ($i = 1; $i <= 10; $i++): ?>
   <P>...</P>
<?php endfor ?>
```

While

```
$i =0;
while ($i < 10)
{
    echo "i vaut $i<br/>";
    $i++;
}
```

i. Fonction

```php
<?php
function SayHello($name)
{
    echo 'Bonjour ' . $name . ' !';
}
SayHello('Jerome');
?>
```

Valeur par défaut

```php
function calcul($a, $b=10)
{
}
```

global

Permet d'accéder à une variable globale depuis une fonction

```php
<?php
$myvar = 10;

function myTest() {
    global $myvar;
}
```

j. Tableau

Création

```php
$fruits = Array("Pomme","Poire","Pêche");
```

Ou

```php
$fruits = Array();
$fruits[0] = "Pomme";
$fruits[1] ="Poire";
// …
```

Vérifier l'existence d'une valeur dans le tableau

```php
if(in_array('Banane', $fruits)){

}
```

Pour ajouter … array_push($tableau, $value)

Pour supprimer … unset($tableau[$index])

Afficher rapidement un tableau

```php
print_r($fruits);
```

Parcourir un tableau

```php
foreach($fruits as $fruit)
{
    echo $fruit . '<br />';
}
```

Tableau associatif

```php
$person = array (
    'firstname' => 'Marie',
    'lastname' => 'Bellin',
    'email' => 'mb3@hotmail.com'
);
```

Equivalent à

```php
$person = array ();
$person['firstname'] = 'Marie';
$person['lastname'] = 'Bellin';
$person['email'] = 'mb3@hotmail.com';
```

Vérifier l'existence d'une clé

```php
if(array_key_exists('firstname', $person)){
 echo $person['firstname'];
}
```

Parcourir un tableau associatif

```php
foreach ($person as $key => $value) {
    echo $key.' '.$value.'<br/>';
}
```

Note à partir de PHP 5.4 on peut remplacer « Array() » par []

```php
$myarray =[];
```

k. Paramètres d'URL ($_GET)

Paramètres d'URL stockés dans le tableau **$_GET**

```php
<?php
if(isset($_GET['id'])) {
    echo $_GET['id'];
}
?>

<a href="details.php?id=12345">lien</a>
```

Vérifier la présence d'un paramètre et ensuite utilisation

+ méthodes utiles : urlencode,htmlspecialchars, htmlentities

l. Formulaire

Exemple de formulaire

```php
<form method="post">
    <input type="text" name="email" placeholder="Email" />
    <!--etc-->
    <input type="submit" name="submit" value="Envoyer" />
</form>
```

Attributs de form :

- **method** : « get » (envoi des **informations dans l'URL)** ou « post » (envoi des informations de manière **cachée)**
- **action** : **page appelée lors de l'envoi**. Ne rien indiquer si c'est la même page.

Attributs importants des éléments de formulaire (input, textarea, etc.)

- attribut « **name** » qui est important pour le passage d'information
- l'attribut id peut être pointé par le label par exemple

Vérifier qu'un formulaire a été soumis

```
if (isset($_POST['submit'])) {

}
```

Ou

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {

}
```

Ou

```
if(!empty($_POST)) {

}
```

Validation des informations saisies

Récupérer les paramètres des tableaux $_GET ou $_POST selon la méthode utilisée.

Obtenir la valeur d'un champ

```
$_POST['email']
```

Vérifier la présence d'une valeur pour un champ

```
if(!empty($_POST) && !empty($_POST['username']) && !empty($_POST['password'])){

}
```

On ajoute les erreurs dans un tableau $errors que l'on affiche

Expression régulière

Filtre prédéfini

```
if(empty($_POST['username']) || !preg_match('/^[a-zA-Z0-9_]+$/', $_POST['username'])){
    $errors['username'] = "Votre pseudo n'est pas valide";
}

if(empty($_POST['email']) || !filter_var($_POST['email'], FILTER_VALIDATE_EMAIL)){
    $errors['email'] = "Votre email n'est pas valide";
}

if(empty($_POST['password']) || $_POST['password'] != $_POST['password_confirm']){
    $errors['password'] = "Vous devez rentrer un mot de passe valide";
}
```

Validation avec « **filter_var ()**» et les Filtres prédéfinis pour la validation (entier, url, ip, email,…)

Affichage des erreurs

```
<?php if(!empty($errors)): ?>
<div class="alert alert-danger">
    <p>Vous n'avez pas rempli le formulaire correctement</p>
    <ul>
        <?php foreach($errors as $error): ?>
            <li><?= $error; ?></li>
        <?php endforeach; ?>
    </ul>
</div>
<?php endif; ?>
```

Ou à côté de chaque champ

```
<input type="text" name="username" value="<?php echo $username;?>"><span
class="error">* <?php echo $errors['username'];?></span>
```

Autres méthodes utiles pour la validation :

- Strlen() vérifier la longueur if(strlen($a))<3) …
- is_string(), is_int(), etc.

Exemple simple

```
<?php
$errors = Array();
$email = "";

if (isset($_POST['submit'])) {
    if (!isset($_POST['email']) || empty($_POST['email'])) {
        $errors['email'] = "Email requis";
    }
    // autres validations de champs

    if(empty($errors)){
        // traitement (sauvegarde vers la base de données par exemple),
redirection
        header('Location: index.php');
        exit();
    }
}
?>
<div>
    <?php foreach($errors as $error) { echo $error."<br/>";} ?>
</div>
<form method="post">
    <input type="text" name="email" placeholder="Email" value="<?php
$email ?>" />
    <!--// autres champs-->
    <input type="submit" name="submit" value="Envoyer" />
</form>
```

Affichage des erreurs de validation

Affichage de l'email saisi avant l'envoi

m. Fonction PHP "extract"

Peut être utile pour extraire directement des variables à partir d'un tableau. Exemple avec $_POST extract($_POST) où on obtient direcment $id, $title au lieu de faire $_POST['id'] …

n. Variables « superglobales »

- *$GLOBALS*
- *$_SERVER variables serveur*
- *$_GET paramètres d'URL*
- *$_POST données envoyées par formulaire*
- *$_FILES liste de fichiers envoyés par formulaire*
- *$_COOKIE valeurs de cookies enregistrés sur l'ordinateur du visiteur*
- *$_SESSION variables de session*
- *$_REQUEST*
- *$_ENV variables d'environnement*

0. Session

Documentation

1. En haut de chaque page

```php
<?php
session_start();
?>
```

Vérifier l'état de la session

```php
<?php
if(session_status() == PHP_SESSION_NONE){
    session_start();
}
```

2. Enregistrement d'une variable de session

```php
$_SESSION['email'] = 'mb3@hotmail.com';
```

```php
$_SESSION['auth'] = $user;
```

3. Utilisation dans la page

```php
<?php
if(isset($_SESSION['email'])){
    echo 'Bonjour '.$_SESSION['email'].'!';
}
?>
```

```php
<?php if (isset($_SESSION['auth'])): ?>
    <li><a href="logout.php">Se déconnecter</a></li>
<?php else: ?>
    <li><a href="register.php">S'inscrire</a></li>
    <li><a href="login.php">Se connecter</a></li>
<?php endif; ?>
<h1>Bonjour <?= $_SESSION['auth']->username; ?></h1>
```

4. Détruire une session : page « **logout.php** »

```php
<?php
session_start();
$_SESSION = array();
session_unset();
session_destroy();
$_SESSION['flash']['success'] = 'Vous êtes maintenant déconnecté';
header("Location:index.php");
?>
```

Flash

Afficher un message flash qui ne sera affiché qu'une fois pour informer l'utilisateur.

```
$_SESSION['flash']['success'] = 'Vous êtes maintenant connecté';
$_SESSION['flash']['danger'] = 'Identifiant ou mot de passe incorrecte';
```

Affichage et destruction de la variable de session

```
<?php if(isset($_SESSION['flash'])): ?>
    <?php foreach($_SESSION['flash'] as $type => $message): ?>
        <div class="alert alert-<?= $type; ?>">
            <?= $message; ?>
        </div>
    <?php endforeach; ?>
    <?php unset($_SESSION['flash']); ?>
<?php endif; ?>
```

p. Cookie

Documentation

Il vaut mieux écrire plusieurs cookies et stocker seulement **une variable par cookie**.

Ecriture de cookies

```
<?php
setcookie('username', 'Marie', time() + 365*24*3600, null, null, false,
true);
// autre cookie

?>
```

« http only »

Note : pour modifier un cookie on utilisera « setcookie » qui écrasera le cookie existant.

Lecture d'un cookie, dans la page

```
<?php
if(isset($_COOKIE['username'])){
    echo 'Bonjour '.$_COOKIE['username'].'!';
}
?>
```

q. Redirection

```
header("Location: index.php");
exit;
```

On peut créer une fonction dédiée

```
function redirect_to($new_location) {
    header("Location: " . $new_location);
    exit;
}
```

r. Inlude et require

include

Sert à insérer le contenu d'un fichier (vue,

```
<?php include('header.php') ?>
<?php include_once('member.php') ?>
```

require

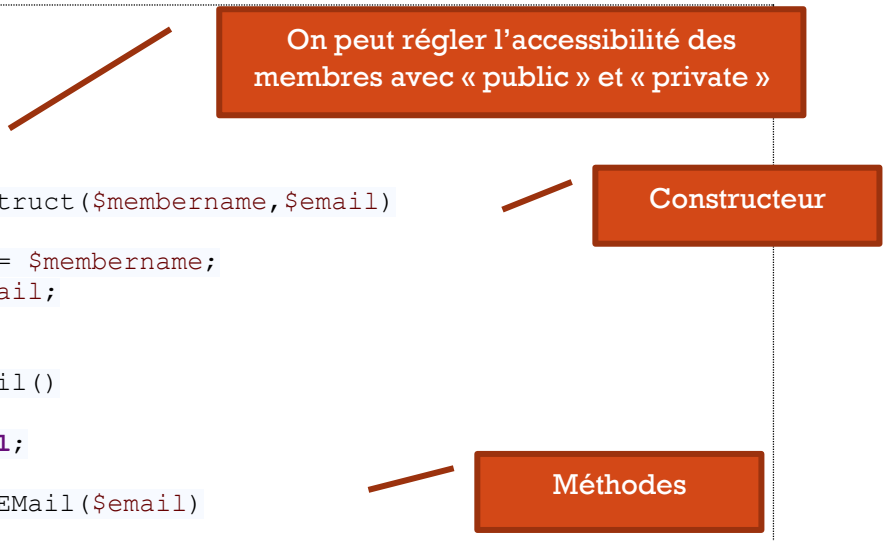
```
<?php require('functions.php') ?>
<?php require_once('functions.php') ?>
```

s. Classes, Objets

Documentation

Base

```php
<?php
class Member
{
    private $email;
    public $membername;

    public function __construct($membername,$email)
    {
        $this->membername = $membername;
        $this->email = $email;
    }

    public function getEmail()
    {
        return $this->email;
    }
    public function changeEMail($email)
    {
        $this->email = $email;
    }
}
?>
```



Utilisation

```php
<?php
require('member.php');
$marie = new Member('Marie Bellin','mb3@hotmail.com');
?>
```

Et dans la page

```php
<h2><?php echo $marie->membername ?></h2>
```

Détruire un objet

```php
unset($marie);
```

Constructeur par défaut

```php
public function __construct()
{
    $this->membername ="";
}
```

Héritage

```php
<?php
require('member.php');

class Admin extends Member
{
}
?>
```

On peut redéfinir les fonctions du parent et faire appel au constructeur du parent avec

```php
public function __construct(){
    parent::__construct();
}
```

Membres static

On peut créer des propriétés et methods “static” et accéder aux methods non static de la classe avec self

```php
<?php

class MyClass {
    public static $my_static_member;

    public function getMessage(){
        return "Hello";
    }

    public static function myStaticFunction(){
        return self::getMessage();
    }
}
```

Singleton

```php
class MySingleton
{
    // ... propriétés de la classe
    private static $_instance;

    public static function getInstance(){
        if(is_null(self::$_instance)){
            self::$_instance = new MySingleton();
        }
        return self::$_instance;
    }

    public function __construct(){
        // initalisation des propriétés $this-> ....
    }
}
```

Namespaces

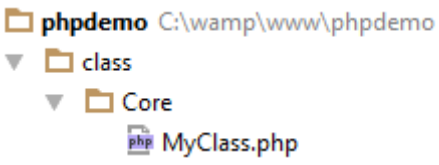
Permettent d’éviter les conflits dans les noms de classes.

Pour définir un namespace

```php
<?php

namespace MyProject\Core;

class MyClass
{
```

Correspond à un nom choisi pour le projet

Puis doit suivre la structure des dossiers



Utilisation :

```php
<?php

use MyProject\Core\MyClass;

$my_instance = new MyClass();
```

Ou sans “use”

```php
$my_instance = new \MyProject\Core\MyClass();
```

Note il faudra placer “\” devant pour l’utilisation de certaines classes de PHP. Exemple avec les dates.

```php
<?php
namespace MyProject\Core;
class MyClass
{
    public function getDate(){
        return new \DateTime();
    }
}
```

Interfaces

```php
<?php

namespace MyProject\Core;

interface MyInterface
{
    public function doSomething();
}

class MyClass implements MyInterface
{
```

t. Exception

```php
if($myvar) {
    throw new Exception("Message");
}
```

try catch

```php
try {
    //...
}
catch(Exception $e) {
    echo 'Message: ' .$e->getMessage();
}
```

u. Trait

Disponible depuis PHP 5.4. C’est un mécanisme permettant d’ajouter du code à une classe plutôt que d’utiliser de l’héritage qui ne serait pas forcément adapté

Documentation

v. Design patterns

Documentation

5. MySQL

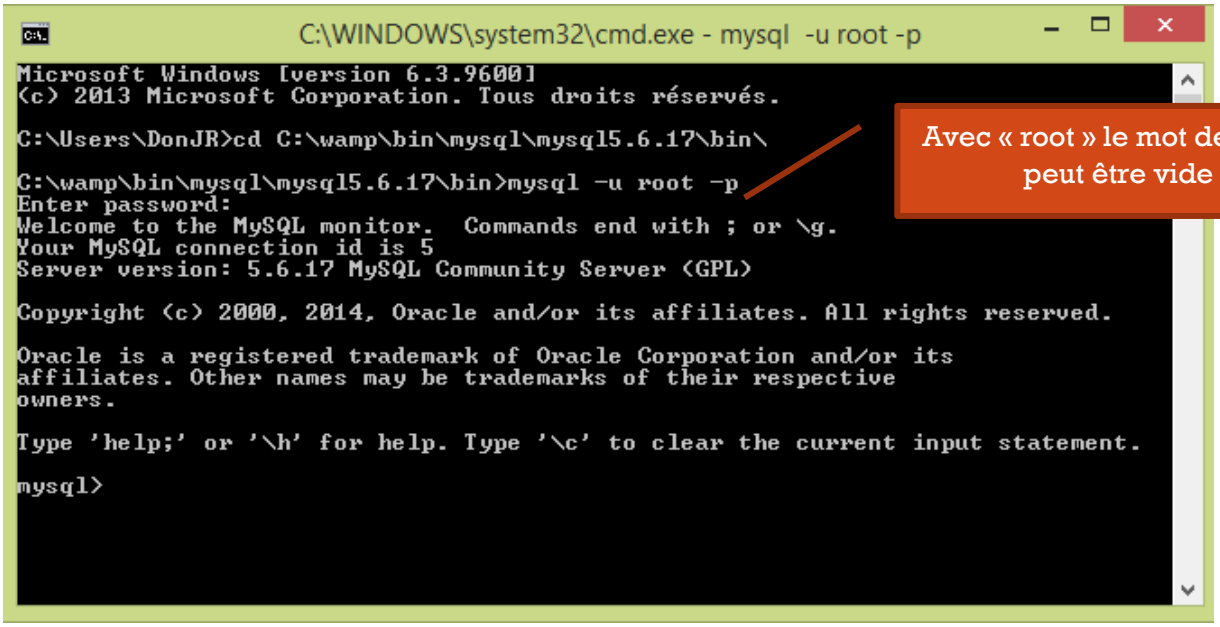
Documentation MySQL

Création/ Gestion de bases de données …

➔ Depuis une **Invite de commande**

Naviguer jusqu'au dossier de mysql. Exemple avec Wamp dossier « C:\wamp\bin\mysql\mysql5.6.17\bin » … puis connexion

```
mysql -u root -p
```



A partir de là il est possible de créer, supprimer des bases de données, des tables, gérer les utilisateurs et droits, etc. **en ligne de commande** … Le « ; » marque la fin d'une instruction et son exécution.

Databases

- « show databases ; » afficher les bases de données
- « create database db_name ; » création d'une base
- « use db_name » switcher sur une base pour les requêtes
- « drop database db_name ; »

Tables

- « create table table_name (…. »
- « show columns from table_name ; »
- « alter table table_name … »
- « drop table table_name ; »

Utiliser la documentation

➔ Soit avec **PhpMyAdmin**

➔ MySQL Workbench permet de créer des schémas de ses bases avec « reverse engineer » et ensuite mettre à jour avec un « forward »

a. Mysqli

Documentation

Créer un service dédié à l'interaction avec une base de données peut éviter d'avoir le code mélangé dans la page (« ArticleService.php » par exemple ou à la rigueur dans « functions.php »).

Connexion

Procédural

```
define("DB_SERVER", "localhost");
define("DB_USER", "root");
define("DB_PASSWORD", "");
define("DB_NAME", "mydb");

function get_connection(){
    $db = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_NAME);
    if(mysqli_connect_errno()) {
        die("Echec lors de la connexion à MySQL :
".mysqli_connect_error()." (" . mysqli_connect_errno(). ")");
    }
    return $db;
}
```

Fermer la connexion

```
mysqli_close($db);
```

Lecture

On crée une fonction

```
function get_articles()
{
    $db = get_connection();
    $result = mysqli_query($db,"SELECT * FROM articles");
    close_connection($db);
    return $result;
}
```

Dans la page affichant les données

```
<?php require_once("article_service.php"); ?>
<?php
 $articles = get_articles();
?>
```

On inclut le fichier contenant les fonctions

On récupère le résultat

On affiche dans la page

```
<?php while ($article =  mysqli_fetch_assoc($articles)): ?>
    <article>
        <h2><?php echo $article['Title']; ?></h2>
        <p><?php echo $article['Content']; ?></p>
    </article>
<?php endwhile ?>
```

Méthodes de lecture :

- « **mysqli_fetch_row** » : résultat dans un tableau, clés sont des entiers
- « **mysqli_fetch_assoc** » : résultat dans un tableau associatif, clés avec noms de colonnes
- « **mysqli_fetch_array** » : résultat dans un tableau ou tableau associatif selon le type MYSQL_NUM, MYSQL_ASSOC, MYSQL_BOTH

Libérer les résultats

```php
<?php
  mysqli_free_result($articles);
?>
```

Ecriture

Fonction d'ajout

```php
function add_article($title,$content)
{
    $db = get_connection();
    $result = mysqli_query($db,"INSERT INTO articles(Title,Content)
VALUES ('{$title}','{$content}')");
    return $result;
}
```

Retourne « true » ou « false » selon la réussite ou l'échec de la requête

Problème des « ' » et **SQL injection** dans les paramètres

- Utiliser « **mysqli_real_escape_string** »

```php
$content = mysqli_real_escape_string($db,$content);
// $result = mysqli_query(…
```

- Méthode « **addslashes** » ajoute automatiquement les « \' »

```php
addslashes($content);
```

- « **Requête préparée** »

```php
function add_article($title,$content)
{
    $db = get_connection();
    $query = "INSERT INTO articles(Title,Content) VALUES (?,?)";
    $stmt = mysqli_prepare($db,$query);
    mysqli_stmt_bind_param($stmt, "ss",$title,$content);
    $result = mysqli_stmt_execute($stmt);
    return $result;
}
```

Type des variables : s (string),i (entier), d (decimal), b (blob)

+ mysqli_stmt_bind_result(…), mysqli_stmt_fetch($stmt) pour les requêtes en lecture et mysqli_stmt_close($stmt)

Page d'ajout d'article

```php
<?php require_once("article_service.php"); ?>
<?php
$message ="";
if (isset($_POST['submit'])) {
    $title = $_POST['title'];
    $content = $_POST['content'];

    $result = add_article($title,$content);
    if($result){
        header("Location: article_added.php");
        exit;
    }
    else {
        $message ="Echec lors de l'ajout de l'article.";
    }
}
?>
<!DOCTYPE html>
<html>
<head lang="en">
    <meta charset="UTF-8">
    <title></title>
</head>
<body>
<h2>Ajouter un article</h2>
<form method="post">
    <input type="text" name="title" /><br/>
    <textarea name="content"></textarea><br />
    <input type="submit" name="submit" value="Ajouter"/>
</form>

<p><?php echo $message ?></p>
</body>
</html>
```

Après l'envoi du formulaire on récupère les variables puis fait appel au service pour ajouter un article dans la base de données. Si l'ajout a réussi on redirige vers une page de succès

Obtenir l'**id** (auto incrémenté) du **dernier élément** ajouté (documentation)

```php
$id = mysqli_insert_id($db);
```

En « Objet »

Connexion

```php
define("DB_SERVER", "localhost");
define("DB_USER", "root");
define("DB_PASSWORD", "");
define("DB_NAME", "mydb");

function get_connection(){
    $db = new mysqli(DB_SERVER, DB_USER, DB_PASSWORD, DB_NAME);
    if ($db->connect_errno) {
        echo "Echec lors de la connexion à MySQL : (" . $db-
>connect_errno . ") " . $db->connect_error;
    }
    return $db;
}
```

Lecture

```php
function get_articles()
{
  $db = get_connection();
  $result = $db->query("SELECT * FROM articles");
   return $result;
}
```

Dans la page

```php
<?php while ($article = $articles->fetch_assoc()): ?>
    <article>
        <h2><?php echo $article['Title']; ?></h2>
        <p><?php echo $article['Content']; ?></p>
    </article>
<?php endwhile ?>
```

Ecriture

```php
function add_article($title,$content)
{
    $db = get_connection();
    $content = mysqli_real_escape_string($db,$content);
    $result = $db->query("INSERT INTO articles(Title,Content) VALUES
('{$title}','{$content}')");
    return $result;
}
```

b. PDO

Documentation

Connexion

```
function get_connection(){
    $db = new PDO('mysql:host=localhost;dbname=mydb;charset=utf8',
'root', '');
    return $db;
}
```

Libérer le résultat

```
$articles->closeCursor();
```

Fermer la connexion

```
$db= null;
```

Lecture

```
function get_articles()
{
    $db = get_connection();
    $result = $db->query("SELECT * FROM articles");
    return $result;
}
```

Dans la page

```
<?php require_once("ArticleService.php"); ?>
<?php
$articles = get_articles();
?>
```

Utilisation du résultat

```
<?php while ($article = $articles->fetch()): ?>
    <article>
        <h2><?= $article['Title']; ?></h2>
        <p><?= $article['Content']; ?></p>
    </article>
<?php endwhile ?>
```

Ecriture

Avec « requête préparée »

```
function add_article($title,$content)
{
    $db = get_connection();
    $stmt = $db->prepare("INSERT INTO articles(Title,Content) VALUES
(:title,:content)");
    $stmt->bindParam(':title', $title);
    $stmt->bindParam(':content', $content);

    $result = $stmt->execute();
    return $result;
}
```

6. Authentification

Service « AccountService »

```php
<?php

define("DB_SERVER", "localhost");
define("DB_USER", "root");
define("DB_PASSWORD", "");
define("DB_NAME", "mydb");

function get_connection(){
    $db = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD, DB_NAME);
    if(mysqli_connect_errno()) {
        die("Echec lors de la connexion à MySQL : ".mysqli_connect_error()." (" .
mysqli_connect_errno(). ")");
    }
    return $db;
}
function close_connection($db){
    if(isset($db)){
        mysqli_close($db);
    }
}
function find_user($email){
    $db = get_connection();
    $sql = "SELECT * FROM users WHERE Email='{$email}'";
    $result = mysqli_query($db, $sql);
    $user = mysqli_fetch_assoc($result);
    close_connection($db);
    return $user;
}
function register($email,$password){
    $db = get_connection();
    $hash = password_hash($password, PASSWORD_DEFAULT);
    $sql = "INSERT INTO users(Email,PasswordHash) VALUES ('{$email}','{$hash}')";
    $result = mysqli_query($db, $sql);
    close_connection($db);
    return $result;
}
function password_check($password,$hash){
   return password_verify($password, $hash);
}
?>
```

« validation_functions »

```php
<?php

function has_presence($value) {
    return isset($value) && (!empty($value));
}

?>
```

a. Inscription

```php
<?php
session_start();

require_once('../services/validation_functions.php');
require_once('../services/account_service.php');

$errors = Array();
$email = "";

if (isset($_POST['submit'])) {
    $email = $_POST['email'];
    $password = $_POST['password'];
    // validation
    if(!has_presence($_POST['email'])){
        $errors['email'] = "Email requis";
    }
    if(!has_presence($_POST['password'])){
        $errors['password'] = "Mot de passe requis";
    }
    else if ($_POST['password']!= $_POST['confirmpassword']) {
        $errors['password'] = "Les mots de passe ne correspondent pas.";
    }
    if(empty($errors)){
        // verifier qu'un utilisateur avec cet email n'existe pas
        $user = find_user($email);
        // register
        if(!$user) {
            $result = register($email,$password);
            if($result){
                // redirection
                header('Location: login.php');
                exit();

            }else {
                $errors['Register'] = "Erreur durant l'inscription.";
            }
        }
        else {
            $errors['Register'] = "Cet email est déjà utilisé.";
        }
    }
}
?>
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title></title>
</head>
<body>
<h2>Inscription</h2>
<form method="post">
    <div>
        <?php foreach($errors as $error) { echo $error."<br/>";} ?>
    </div>
    <div>
        <input type="email" name="email" placeholder="Email" value="<?php
echo $email; ?>" autofocus/>
    </div>
    <div>
```

```
        <input type="password" name="password" placeholder="Mot de passe"
/>
    </div>
    <div>
        <input type="password" name="confirmpassword"
placeholder="Confirmer le mot de passe" />
    </div>
    <div>
        <input type="submit" name="submit" value="Inscription">
    </div>
</form>
</body>
</html>
```

Table utilisée

```
CREATE TABLE IF NOT EXISTS `users` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `Email` varchar(255) NOT NULL,
  `PasswordHash` varchar(255) NOT NULL,
  `Created` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`Id`),
  UNIQUE KEY `Email` (`Email`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;
```

b. Logout

Page vers laquelle se diriger pour détruire la session

```
<?php
session_start();
$_SESSION = array();
session_unset();
session_destroy();
header("Location:../index.php");
?>
```

c. Login

```php
<?php
session_start();

require_once('../services/validation_functions.php');
require_once('../services/account_service.php');

$errors = Array();
$email = "";

if (isset($_POST['submit'])) {
    $email = $_POST['email'];
    $password = $_POST['password'];
    // validation
    if(!has_presence($_POST['email'])){
        $errors['email'] = "Email requis";
    }
    if(!has_presence($_POST['password'])){
        $errors['password'] = "Mot de passe requis";
    }
    // verification
    if(empty($errors)){
        // retrouver l'utilisateur
        $user = find_user($email);
        if ($user) {
            $hash = $user['PasswordHash'];
            if (password_check($password, $hash)) {
                // session
                $_SESSION['Email'] = $email;
                // redirection
                header("Location:../index.php");
                exit();

            } else {
                $errors['Login'] = "Email ou mot de passe non trouvé.";
            }
        } else {
            $errors['Login'] = "Pas d'utilisateur avec cet email.";
        }
    }
}
?>
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>Document sans titre</title>
</head>
<body>
<form method="post">
    <div>
        <?php foreach($errors as $error) { echo $error."<br/>";} ?>
    </div>
    <input type="text" name="email" placeholder="Email" />
    <input type="password" name="password" placeholder="Mot de passe" />
    <input type="submit" name="submit" value="Connexion" />
    <a href="register.php">Pas encore inscrit?</a>
</form>
</body>
</html>
```

7. Envoi de mail

Envoi de mail en local avec Wamp. Configuration de **PHP.ini**

Configuration

```
[mail function]
; For Win32 only.
; http://php.net/smtp
SMTP = smtp.orange.fr
; http://php.net/smtp-port
smtp_port = 25

; For Win32 only.
; http://php.net/sendmail-from
sendmail_from =                @wanadoo.fr
```

Exemple envoi de mail au format html

Documentation

```
$headers = 'Content-type: text/html; charset=iso-8859-1';
$link="<a href='http://romagny13.com'>lien</a>";
mail('romagny13@yahoo.fr', 'Titre',"Message avec $link .",$headers);
```

8. Gravatar

Gravatar est un service utilize avec Wordpress. Il est toutefois possible de récupérer son image pour son simplement en PHP.

On a besoin de :

- L'**email** que l'on passe en le cryptant en md5
- La **taille** de l'**image**

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .avatar{
            border-radius: 50%;
        }
    </style>
</head>
<body>
<?php
$email ="romagny13@yahoo.fr";
$size = 200;
$avatar= 'http://2.gravatar.com/avatar/'. md5($email). '?s='. $size;
?>

<img src="<?= $avatar ?>" class="avatar">

</body>
</html>
```

9. URL Rewriting

Documentation Apache

Le module "**rewrite_module**" d'**Apache** doit être active pour la réécriture d'url

On ajoute un fichier « **.htaccess** » à la racine du projet.

Exemple d'url que l'on pourrait imaginer pour un site MVC

```
RewriteEngine On

RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(\w+)\/((\w+)\/)?([0-9]+)$ index.php?c=$1&a=$3&id=$4 [QSA,L]
```

Condition, Ne pas réécrire si le fichier existe

Flags

Lorsque l'on saisit "http://localhost/urlrewrite/moncontroller/monaction/10" en fait cela ne récupère que "**moncontroller/monaction/10**" pour la réécriture.

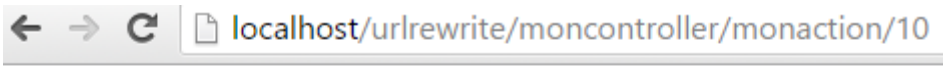
Ainsi avec les expressions régulières on peut isoler par groupes (entre parentheses) ce que l'on veut extraire. On peut tester ses regex.

Groupes entre parentheses:

- ➔ $1 (\w+)
- ➔ $2 ((\w+)\/)? L'avantage de ce groupe ici, c'est que l'on pourra avoir ou non une action
- ➔ $3 (\w+) à « l'intérieur » de $2
- ➔ $4 ([0-9]+)

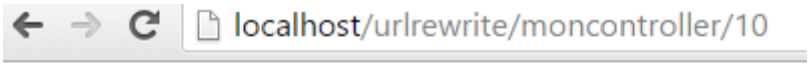
…puis ce que l'on veut obtenir « **index.php?c=$1&a=$3&id=$4** »

- Place le contenu du groupe $1 dans c
- Le contenu de groupe $3 dans a
- Le contenu du groupe $4 dans id




```
array (size=3)
  'c' => string 'moncontroller' (length=13)
  'a' => string 'monaction' (length=9)
  'id' => string '10' (length=2)
```

… sans action




```
array (size=3)
  'c' => string 'moncontroller' (length=13)
  'a' => string '' (length=0)
  'id' => string '10' (length=2)
```

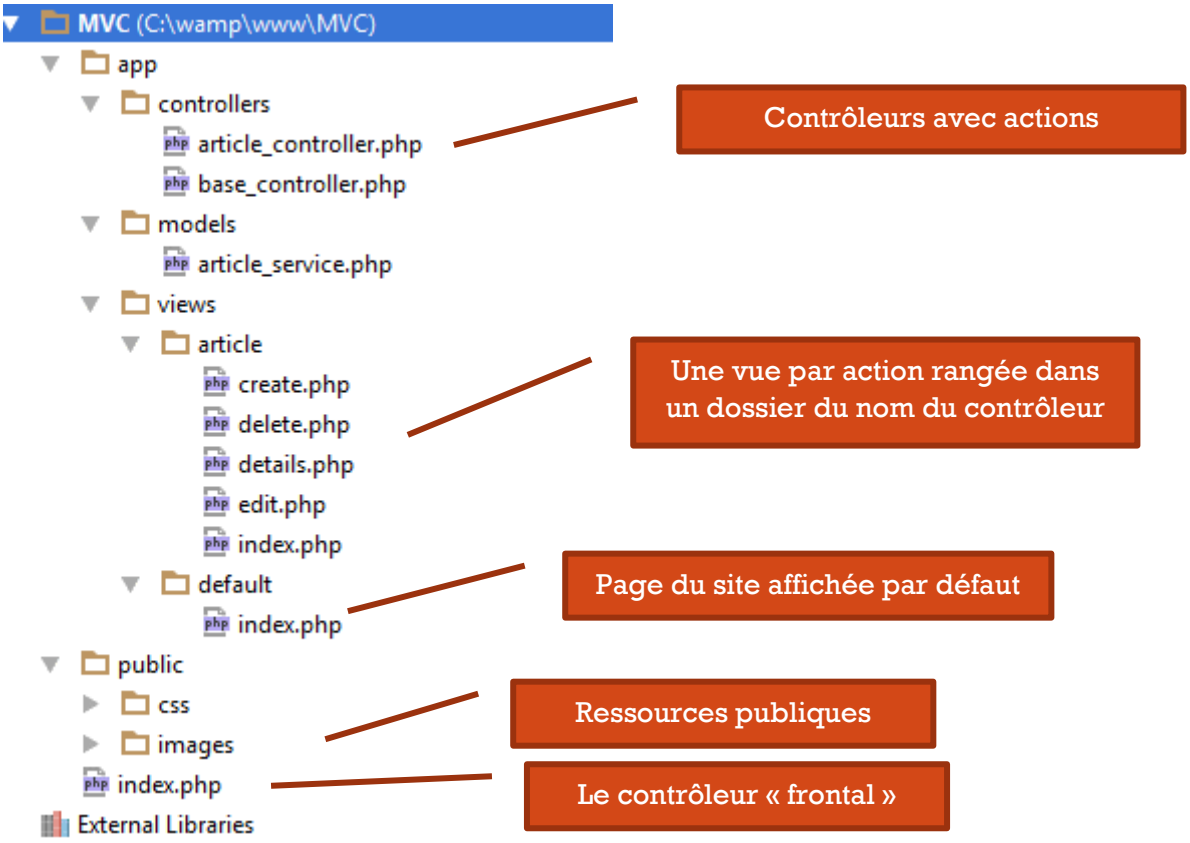
On peut obtenir depuis le PHP la valeur de c,a et id avec $_GET

Exemple : `$id =$_GET['id'];`

Si on veut extraire tout et placer par exemple dans une variable "url"

```
RewriteEngine on

RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(.*)$ index.php?url=$1 [QSA,L]
```

10. MVC « simple »



Routes :

- « http://localhost/MVC/index.php?c=article »
- « http://localhost/MVC/index.php?c=article&a=details&id=2 »

➔ c pour le contrôleur
➔ a pour l'action
➔ id pour le paramètre d'url par exemples

« index.php » récupère les informations de route et redirige vers le bon contrôleur et l'action

```php
<?php
include_once './app/controllers/base_controller.php';
include_once './app/controllers/article_controller.php';

$controller = isset($_GET['c']) ? $_GET['c'] : '';
$action = isset($_GET['a']) ? $_GET['a'] : '';
$id     = isset($_GET['id']) ? $_GET['id'] : '';

// controleurs
switch($controller) {
    case 'article':
        $controller = new ArticleController();
        break;
    default:
        $controller = new BaseController();
}
// action
$controller->run($action, $id);
```

Récupération des informations de

Création du contrôleur

Action

« base_controller.php »

```php
<?php

class BaseController
{
    public $model = null;

    public function run($action = 'index', $id = 0)
    {
        if (!method_exists($this, $action)) {
            $action = 'index';
        }
        return $this->$action($id);
    }
    public function index()
    {
        include 'app/views/default/index.php';
    }
}
```

Contrôleur « article_controller.php »

```php
<?php

class ArticleController extends BaseController{

    public function index()
    {
        // récupération de la liste des articles
        require_once('app/models/article_service.php');
        $articles = get_articles();
        // affichage
```

```
            include 'app/views/article/index.php';
        }
        public function details($id)
        {
            include 'app/views/article/details.php';
        }
        public function create()
        {
            require_once('app/models/article_service.php');
            $message ="";
            if (isset($_POST['submit'])) {
                $title = htmlentities($_POST['title']);
                $content = htmlentities($_POST['content']);

                $result = add_article($title,$content);
                if($result){
                    header("Location: index.php?c=article");
                    exit;
                }
                else {
                    $message ="Echec lors de l'ajout de l'article.";
                }
            }
            include 'app/views/article/create.php';
        }
        // ect.
}
```

Exemple de vue : vue affichant tous les articles (index.php du contrôleur article)

```
<!DOCTYPE html>
<html>
<head lang="en">
    <meta charset="UTF-8">
    <title></title>
    <link rel="stylesheet" type="text/css" href="public/css/default.css">
</head>
<body>
<a href="index.php?c=article&a=create">Ajouter un article</a>
<h2>Liste des articles</h2>
<?php while ($article =  mysqli_fetch_assoc($articles)): ?>
    <article>
        <h2><a href="index.php?c=article&a=details&id=<?php echo
$article['Id'] ?>" ><?php echo $article['Title']; ?></a></h2>
        <p><?php echo $article['Content']; ?></p>
    </article>
<?php endwhile ?>
<?php
  mysqli_free_result($articles);
?>
</body>
</html>
```

Lien vers le détail

Avec réécriture d'URL

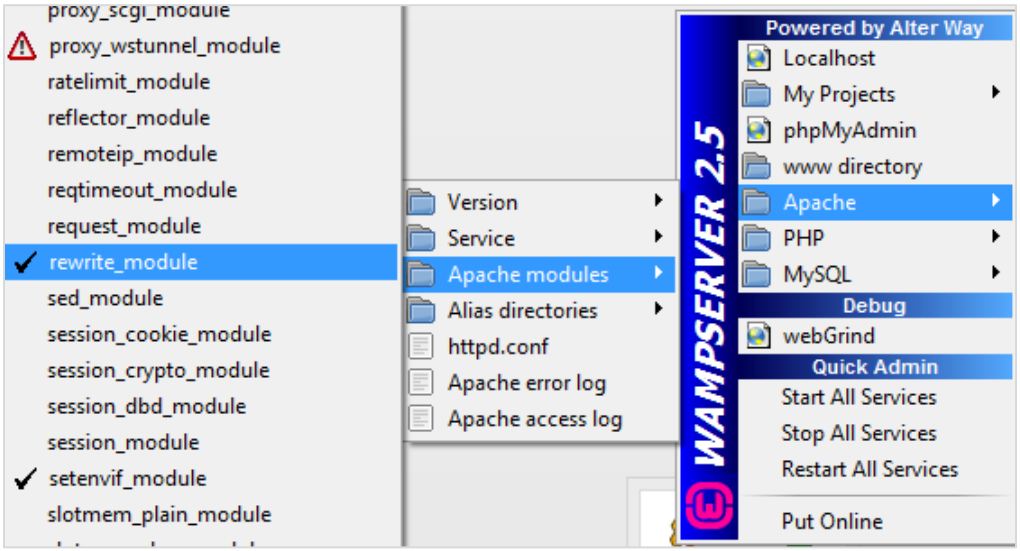
L'intérêt ici est de ne plus avoir des url de la forme :

```
http://localhost/MVC/index.php?c=article&a=details&id=2
```

… Mais

http://localhost/MVC/article/details/2

1. Activer le module « **rewrite_module** » d'**Apache**. Exemple avec « Wamp »



2. Ajout d'un fichier « **.htaccess** » à la racine du projet (on pourrait également mettre la page d'accueil du site et .htaccess dans le dossier « public »)

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^ index.php [L]
```

3. Front controller « index.php »

```
<?php
include_once './app/controllers/base_controller.php';
include_once './app/controllers/article_controller.php';

$url= $_SERVER['REQUEST_URI'];
$url = trim($url, '/');
$url_params = explode('/', $url);
$controller = isset($url_params[1]) ? $url_params[1] : 'default';
$action = isset($url_params[2]) ? $url_params[2] : 'index';
$id = isset($url_params[3]) ? $url_params[3] : '';
//$params = array_slice($url_params, 2);

// controleurs
switch($controller) {
    case 'article':
        $controller = new ArticleController();
        break;
    default:
        $controller = new BaseController();
}
```

Commencer l'index à 0 (http://monsite.com/) ou 1 (http://monsite.com/blog/) selon l'url du

On pourrait récupérer un tableau de paramètres s'il y en avait plusieurs

```php
// action
$controller->run($action, $id);
```

Modifier également tous les liens dans les vues. Exemple

```
<a href="./article/create">Ajouter un article</a>
```

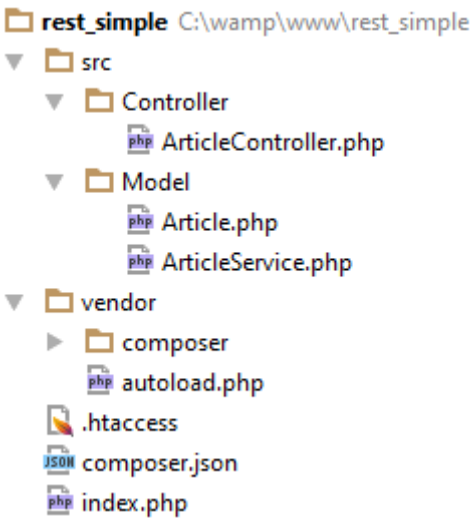
11. Service REST simple

De la même manière on pourrait se créer un service REST

Le but étant d'avoir par exemple

```
http://local-rest.dev/articles/
http://local-rest.dev/articles/2
```

Exemple de structure



Pour l'exemple on peut utiliser **Composer** et son **Autoloader**

```
composer init
```

On définit le répertoire src

```
{
     "name": "romag/rest_simple",
     "license": "MIT",
      "autoload": {
         "psr-4": {
             "App\\": "src"
         }
     },
     "require": {}
}
```

Puis

```
composer install
```

« index.php »

```php
<?php

require 'vendor/autoload.php';

// Sans Autoloader
//require 'src/Controller/ArticleController.php';
//require 'src/Model/ArticleService.php';
//require 'src/Model/Article.php';
header("Access-Control-Allow-Origin: *");
header('Access-Control-Allow-Methods: *');
header('Access-Control-Allow-Headers: *');

$url = $_SERVER['REQUEST_URI'];
$method = $_SERVER['REQUEST_METHOD']; // GET, POST, PUT, DELETE ..
$url = trim($url, '/');
$split = explode('/', $url);
$route_controller = isset($split[0]) ? $split[0] : 'default';
$id = isset($split[1]) && preg_match('/^[0-9]+$/', $split[1]) ? $split[1] : '';

switch ($route_controller) {
    case 'articles':
        $controller = new App\Controller\ArticleController();
        if ($method == "GET") {
            if ($id != "") {
                $controller->getOne($id);
            } else {
                $controller->getAll();
```

```
            }
        }
        if ($method == "POST") {
            $controller->add();
        }
        if ($method == "PUT") {
            $controller->update();
        }
        if ($method == "DELETE") {
            $controller->delete($id);
        }

        break;
    default:
}
```

Le contrôleur

```
<?php

namespace App\Controller;

use App\Model\ArticleService;
use App\Model\Article;

class ArticleController
{
    public $articleService;

    public function __construct()
    {
        $this->articleService = new ArticleService();
    }

    public function getAll()
    {
        $articles = $this->articleService->getAll();
        echo json_encode($articles);
    }

    public function getOne($id)
    {
        $article = $this->articleService->getOne($id);
        echo json_encode($article);
    }

    public function add()
    {
        $contentType =  $_SERVER["CONTENT_TYPE"];

        if($contentType === "application/x-www-form-urlencoded"){
            //var_dump($_POST);
            $newArticle = new Article(0, $_POST['title'],$_POST['content']);
            $result = $this->articleService->add($newArticle);
            echo json_encode($result);
        }
        else if($contentType === "application/json"){
            $json = file_get_contents('php://input');
            $data = json_decode($json);
            $result = $this->articleService->add($data);
```

```
            echo json_encode($result);
        }
        else{
            echo json_encode(array("error" => "Pas de données reçues"));
        }
    }

    public function update()
    {
        if($_SERVER["CONTENT_TYPE"] === "application/json"){
            $json = file_get_contents('php://input');
            $data = json_decode($json);

            $result = $this->articleService->update($data);
            echo json_encode($result);
        }
    }

    public function delete($id){
        $this->articleService->delete($id);
    }
}
```

Le « fake » service

```
<?php

namespace App\Model;

class ArticleService
{
    public static $articles = null;

    public function __construct()
    {
        self::$articles = Array();
        self::$articles[0] = new Article(1, "Premier article", "Lorem ipsum ...");
        self::$articles[1] = new Article(2, "Second article", "Lorem ipsum atum ...");
    }

    public function getAll()
    {
        return self::$articles;
    }

    public function getOne($id)
    {
        $index = $this->getIndex($id);
        if($index !== -1){
            return self::$articles[$index];
        }
        return null;
    }
```

```
    public function add($article)
    {
        $article->id = self::getUniqueId();
        array_push(self::$articles, $article);
        return $article;
    }

    public function update($article)
    {
        $index = $this->getIndex($article->id);
        if($index !== -1){
            $toUpdate = self::$articles[$index];
            $toUpdate->title = $article->title;
            $toUpdate->content = $article->content;
            return $toUpdate;
        }
        return null;
    }

    public function delete($id)
    {
        $index = $this->getIndex($id);
        if($index !== -1){
            unset(self::$articles[$index]);
            return 1;
        }
        return 0;
    }

    private function getIndex($id){

        for ($i = 0; $i <= count(self::$articles); $i++){
            if(self::$articles[$i]->id == $id){
                return $i;
            }
        }
        return -1;
    }
    private function getUniqueId()
    {
        $count = count(self::$articles) - 1;
        $lastId = self::$articles[$count]->id;
        return $lastId + 1;
    }
}
```

Et le modèle

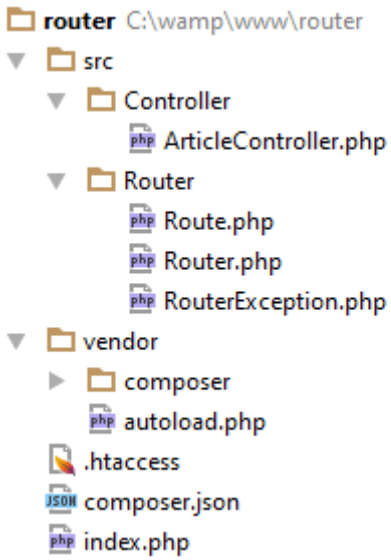
```
<?php

namespace App\Model;

class Article
{
    public $id;
    public $title;
    public $content;

    public function __construct($id, $title, $content)
    {
        $this->id = $id;
```

```
            $this->title = $title;
            $this->content = $content;
        }
}
```

Pour tester le service on peut utiliser par exemple l'extension Chrome **DHC ou Fiddler**

12. Router

Inspiré par le tutoriel de Grafikart.



composer init

```
{
    "name": "romag/router",
    "autoload": {
        "psr-4": {
            "App\\": "src"
        }
    },
    "require": {}
}
```

On indique le dossier contenant les sources pour l'Autoloader

composer install

.htaccess

```
RewriteEngine on

RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(.*)$ index.php?url=$1 [QSA,L]
```

Index.php

```php
<?php

require 'vendor/autoload.php';

//die($_GET['url']); // voir htaccess pour l'url exemple pour
http://localhost/router/articles/1 renvoie articles/1

$router = new \App\Router\Router($_GET['url']);

// Route de base
$router->get('/',function(){
    echo 'Page d\'accueil';
});

$router->get('/articles',function(){
 echo 'Tous les articles';
});

$router->get('/articles/:id',function($id){
    echo 'Article seul id : ' . $id;
});

// plusieurs paramètres
$router->get('/articles/:id/:x',function($id,$x){
    echo 'Article seul id : ' . $id . ' et x: '.$x ;
});

// Route avec contrainte sur le paramètre
$router->get('/test/:name',function($name){
    echo 'Test avec : ' . $name;
})->with('name','[a-z]+');

// CONTROLLERS
$router->get('/articles/controller','Article@index');
$router->get('/articles/controller/:id','Article@getOne');

// Affichage de formulaire
$router->get('/articles/new',function(){
    //echo 'Ajout d\'un article';
    ?>

    <form method="post" action="http://localhost/router/articles/new">
        <input type="text" name="title">
        <input type="submit" value="Envoyer">
    </form>

    <?php

});

// ajout
$router->post('/articles/new',function(){
    echo 'Ajout article';
    var_dump($_POST);
});

$router->run();
```

Router (Router.php)

```php
<?php

namespace App\Router;

class Router
{
    private $url;
    private $routes = Array();

    public function __construct($url)
    {
        $this->url = $url;
    }

    public function get($path,$callable){
        $route = $this->addRoute($path,$callable,'GET');
        return $route;
    }

    public function post($path,$callable){
        $route = $this->addRoute($path,$callable,'POST');
        return $route;
    }

    public function put($path,$callable){
        $route = $this->addRoute($path,$callable,'PUT');
        return $route;
    }

    public function delete($path,$callable){
        $route = $this->addRoute($path,$callable,'DELETE');
        return $route;
    }

    private function addRoute($path, $callable, $method){
        $route = new Route($path, $callable);
        $this->routes[$method][] = $route;
        return $route;
    }

    public function run(){
//        var_dump($this->routes);
//        die;
        // Récupération de la méthode (GET, POST, PUT, DELETE ..)
        if(!isset($this->routes[$_SERVER['REQUEST_METHOD']])){
            throw new RouterException('REQUEST_METHOD does not exist');
        }
        // exécution de la fonction pour la route
        foreach($this->routes[$_SERVER['REQUEST_METHOD']] as $route){
            if($route->match($this->url)){
                return $route->call();
            }
        }
        throw new RouterException('No matching routes');
    }
}
```

Classe Route (Route.php)

```php
<?php

namespace App\Router;

class Route
{
    private $path; //articles/controller/:id
    private $callable;
    private $parameters = [];
    private $params = [];

    public function __construct($path, $callable){
        $this->path = trim($path, '/');
        $this->callable = $callable;
    }

    public function match($url){
        // vérifier si URL actuelle navigateur correspond à la regex du PATH
        $url = trim($url, '/'); // retire la barre de fin si besoin
        $path = preg_replace_callback('#:([\w]+)#', [$this, 'paramMatch'], $this->path);
        $regex = "#^$path$#i"; // drapeau vérifie la casse
        //var_dump($regex);
        if(!preg_match($regex, $url, $matches)){
            return false;
        }
        array_shift($matches); // supprime le premier élément pour ne pas avoir par exemple articles/1
        var_dump($matches);
        $this->parameters = $matches; // récupère les valeurs de paramètres (exemple 2)
        return true;
    }

    private function paramMatch($match){
        //var_dump($match); // :id et id par exemple
        if(isset($this->params[$match[1]])){
            return  '('. $this->params[$match[1]] . ')';
        }
        return '([0-9]+)';
    }

    public function with($param, $regex){
        $this->params[$param] = str_replace('(', '(?:', $regex);
        return $this;
    }

    public function call(){
        // si le callable est un controller exemple "Article@index"
        if(is_string($this->callable)){
            $params = explode('@', $this->callable); // on obtient Article et index
            $class = "App\\Controller\\" . $params[0] . "Controller";
            $controller = new $class();
            return call_user_func_array(array($controller, $params[1]), $this->parameters);
        }
        else{
            return call_user_func_array($this->callable, $this->parameters);
        }
    }
}
```

Une classe d'exception

```php
<?php
namespace App\Router;

class RouterException extends \Exception {

}
```

Un contrôleur simple

```php
<?php
namespace App\Controller;


class ArticleController
{

    public function index(){
        echo 'Dans index du controleur Article';
    }


    public function getOne($id){
        echo 'Dans controleur Article avec id : '.$id;
    }
}
```

13. Slim

Site , Request, Response

a. Installation

Avec Composer dans la console

```
composer require slim/slim "^3.0"
```

… Ou dans le fichier « **composer.json** » créé après un « **composer init** »

```
{
    "name": "romag/slim-demo",
    "require": {
        "slim/slim" :"^3.0"
    }
}
```

Puis « **composer install** »

.htaccess (à la racine du projet)

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^ index.php [QSA,L]
```

b. Routes

Autoloader de Composer en haut de « index.php »

```
<?php
require 'vendor/autoload.php';
```

Routes de bases

Très simple, page d'accueil

```
<?php
require 'vendor/autoload.php';

$app = new Slim\App();

$app->get("/", function () {
    echo "Page d'accueil";
});

$app->run();
```

Ne pas oublier

On peut également utiliser

```
<?php
require 'vendor/autoload.php';

$app = new Slim\App();

$app->get("/", function ($request, $response, $args) {
    $response->write("Page d'accueil");
    return $response;
});

$app->run();
```

Voir

```php
<?php
require 'vendor/autoload.php';

use \Slim\Http\Request;
use \Slim\Http\Response;

$app = new Slim\App();

$app->get("/",function (Request $request, Response $response){
    $response->write("Page d'accueil");
    return $response;
});

$app->run();
```

Autre exemple

```php
$app->get("/articles", function (Response $response) {
    $response->write("Liste des articles");
    return $response;
});
```

c. Avec paramètre

Ici le paramètre peut être un nombre ou une chaine de caractères

```php
$app->get('/articles/{id}', function (Request $request, Response
$response) {
    $id = $request->getAttribute('id'); // récupération du paramètre

    $response->getBody()->write("Article d'id : $id");
    return $response;
});
```

Autre possibilité

Le paramètre ici est un entier

```php
$app->get('/articles/{id}', function (Request $request, Response
$response, $args) {
    $id = (int)$args['id'];

    $response->getBody()->write("Article d'id : $id");
    return $response;
});
```

Paramètre optionnel

```php
$app->get('/articles[/{id}]', function ($request, $response, $args) {

});
```

Regex

```php
$app->get('/articles/{id:[0-9]+}', function ($request, $response, $args)
{

});
```

d. Récupérer les paramètres d'url

```
// /test?id=10&name=jerome
$app->get("/test", function (Request $request, Response $response) {

    $queryParameters = $request->getQueryParams();
     var_dump($queryParameters);
});
```

POST, PUT , DELETE

```
$app->post('/articles', function ($request, $response, $args) {

});

$app->put('/articles/{id}', function ($request, $response, $args) {

});

$app->delete('/articles/{id}', function ($request, $response, $args) {

});
```

Custom

```
$app->map(['GET', 'POST'], '/articles', function ($request, $response,
$args) {

});
```

+ Any

POST

```
$app->post("/articles",function(Request $request, Response $response){
    $data = $request->getParsedBody();
    $article_data =[];
    $article_data['title'] = filter_var($data['title'],
FILTER_SANITIZE_STRING);
    // etc.
});
```

e. Afficher une vue

```
$app->get("/article", function () {
 require 'templates/article.php';
});
```

Avec php-view

```
composer require slim/php-view
```

```php
<?php
use Slim\Views\PhpRenderer;

require "vendor/autoload.php";

$app = new Slim\App();
$container = $app->getContainer();
$container['renderer'] = new PhpRenderer("./templates");

$app->get('/article', function ($request, $response) {
    return $this->renderer->render($response, "/article.php", [ "message"
=> "hello"]);
});

$app->run();
```

Dans la vue

```php
<h1>Template</h1>
<?= $message ?>
```

On peut également utiliser twig-view

f. JSON

```php
$app->get("/test_json", function (Request $request, Response $response) {
    $data = ["title" => "Le titre", "content" => "Le contenu"];
    $response->withJson($data);
    return $response;
});
```

g. Contrôleur

```php
$app->get("/articles", '\App\Controller\ArticleController:index');
```

Avec paramètre

```php
$app->get("/articles/{id}", '\App\Controller\ArticleController:details');
```

```php
<?php

namespace App\Controller;

class ArticleController
{
    public function index() {
      echo "Depuis le controluer Article";
    }

    public function details($request,$response) {
        $id = $request->getAttribute('id');
        echo "Details de l'article id " . $id;
    }
}
```

h. Route nommée

```php
$app->get('/articles/{id}', function ($request, $response, $args) {

})->setName('article_details');
```

+ group

14. Tests

Tests unitaires avec PHPUnit

Documentation, documentation 5.7

Installation

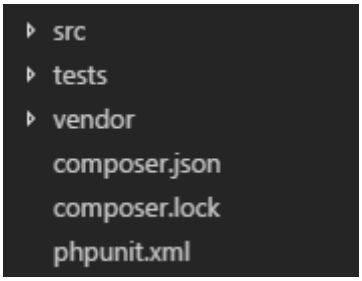
Avec composer

```
composer require phpunit/phpunit --dev
```

Exemple de fichier composer.json

```
{
   "name": "romag/demo",
   "authors": [
        {
            "name": "romagny13",
            "email": "romagny13@yahoo.fr"
        }
   ],
   "autoload":{
        "psr-4":{
            "App\\": "src"
        }
   },
   "require-dev": {
        "phpunit/phpunit": "5.7"
   },
   "require": {}
}
```

Structure de projet



Ecriture de tests

Créer un dossier « **tests** » et y ajouter des classes de tests (exemple : « ArticleServiceTest »). Seules les méthodes commençant par test sont exécutées.

```
<?php


class DemoTest extends PHPUnit_Framework_TestCase
{
    function testMyMethod(){

        // code

        // assertions
        $this->assertTrue(…);
    }
}
```

Assertions

Documentation

Exécution

Terminal ou Powershell avec Windows

```
./vendor/bin/phpunit ./tests
```

Ou depuis une invite de commande, naviguer jusqu'au répertoire « vendor/bin » contenant phpunit. Indiquer en second paramètre le répertoire contenant les tests

```
phpunit ../../tests
```

Fichier de configuration

Documentation

Exemple simple de fichier de configuration

```
<phpunit bootstrap="vendor/autoload.php" colors="true">
  <testsuites>
    <testsuite name="unit">
      <directory suffix=".php">tests</directory>
    </testsuite>
  </testsuites>
  <filter>
    <whitelist>
        <directory>src</directory>
    </whitelist>
  </filter>
</phpunit>
```

Le suffixe permet de filtrer les fichiers à executer. On pourrait également mettre « Test.php » pour des fichiers finissant ainsi

Withelist requise pour le code coverage

Autre exemple

```
<?xml version="1.0" encoding="utf-8" ?>
<phpunit colors="true">
    <testsuite name="MyTests">
        <directory>tests</directory>
    </testsuite>
    <filter>
        <whitelist proceessUncoveredFromWhitelist="true">
            <directory suffix=".php">src/</directory>
        </whitelist>
    </filter>
    <logging>
        <log type="coverage-html" target="build/report" charset="UTF-8"
             highlight="false" lowUpperBound="35" highLowerBound="70"/>
        <log type="testdox-html" target="build/testdox.html"/>
    </logging>
</phpunit>
```

Génération d'un rapport html sur le code couvert

Commande

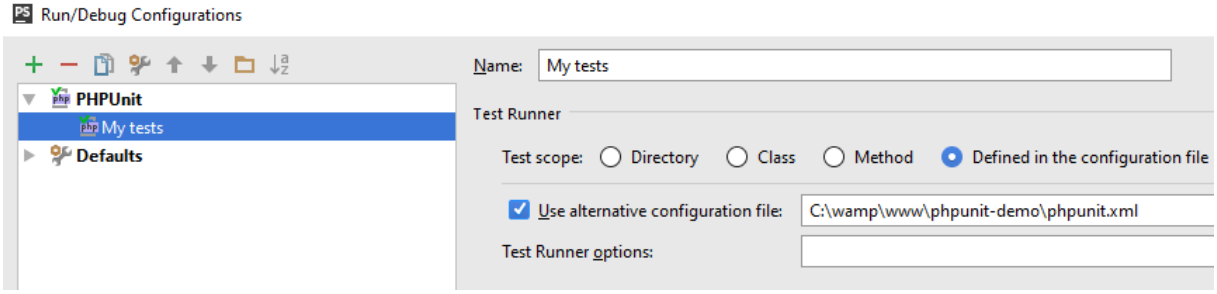
```
phpunit -c .
```

Ou

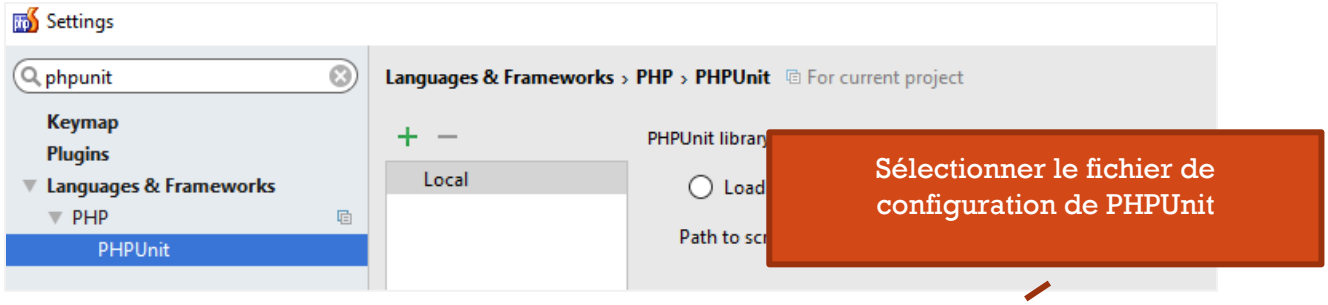
```
phpunit --configuration ../../phpunit.xml --testsuite MyTests
```

Configurer PHPStorm

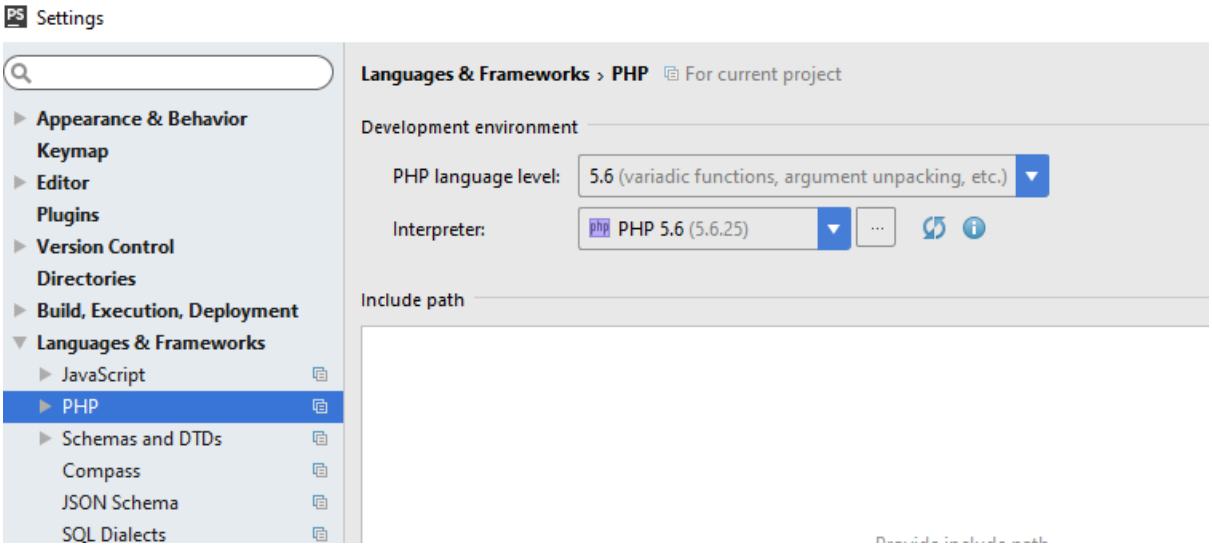
run .. edit configurations .. add phpunit



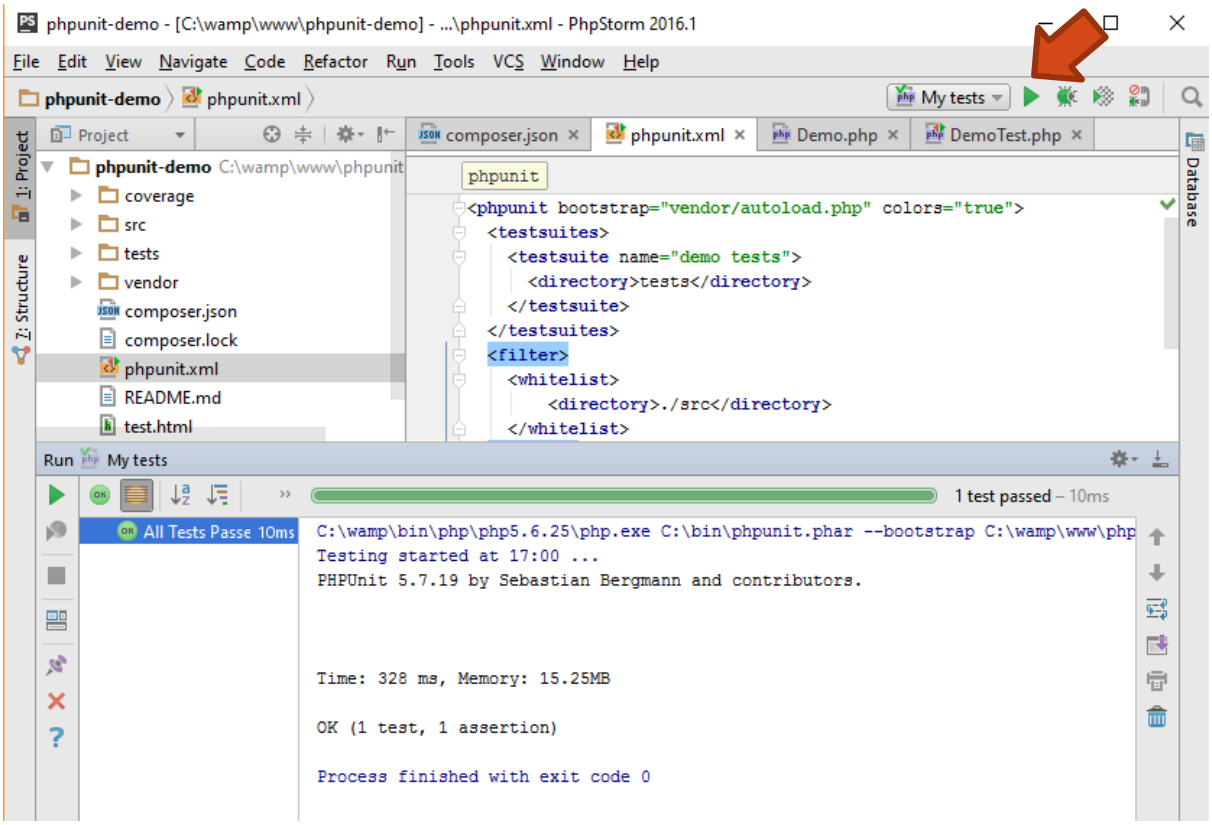
On peut indiquer dans les Settings le chemin vers l'autoload de Composer si besoin



Configurer l'interpréteur si ce n'est pas fait dans les settings



On peut désormais lancer les tests simplement



Code coverage

Créer un dossier nommé par exemple « coverage ». Il suffit de cliquer sur la page d'index

```
phpunit -c . --coverage-html coverage
```

On peut ignorer des méthodes pour la couverture

```
/**
 * @codeCoverageIgnore
 */
public static function fn(){

}
```

Astuce : Activer **xdebug (requis pour le code coverage)** avec **wamp**
Ouvrir php.ini (dossier de la version php utilisée) et ajouter (ou modifier) à la fin

```
; XDEBUG Extension
[xdebug]
zend_extension ="c:/wamp/bin/php/php5.6.25/zend_ext/php_xdebug-2.4.1-5.6-vc11-x86_64.dll"
xdebug.remote_enable = true
xdebug.profiler_enable = 1
xdebug.profiler_enable_trigger = 1
xdebug.profiler_output_name = cachegrind.out.%t.%p
xdebug.profiler_output_dir ="c:/wamp/tmp"
xdebug.show_local_vars=0
```

+ Ajouter une **withe list** au **fichier de config**

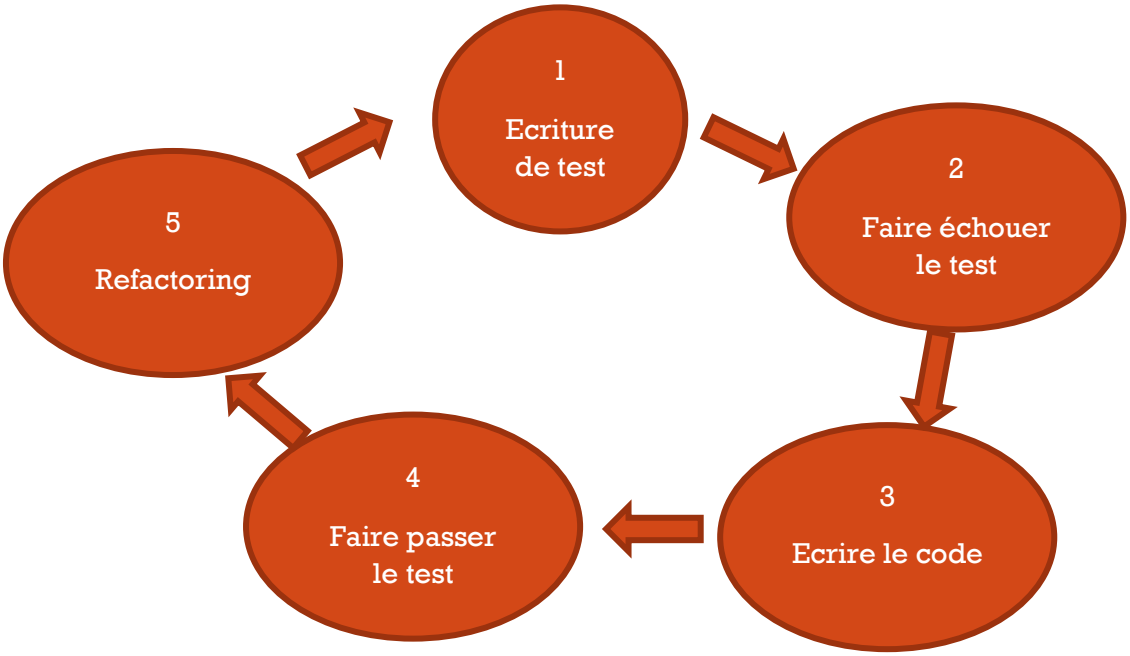
Exécuter un script SQL permettant de réinitialiser une table/ base de données

```php
public function setUp()
{
    exec("mysql -u'root' --password='' < ".__DIR__."/../fixture.sql");
}
```

Le script contient un drop database ou drop table par exemple puis un create table, etc.

Permet de réinitialiser les tests pendant la phase de développement

Intégration continue



Jenkins