# PHP

Les conventions de nommage


# Qu’est-ce que PHP ?

Contrairement au HTML, CSS et JavaScript qui sont des langages utilisés côté client (front-end), PHP (pour Hypertext Preprocessor) est utilisé côté serveur (back-end) pour produire des pages Web dynamiques. Il a été créé en 1994 par Rasmus Lerdof qui voulait se simplifier la conception de sites internet.

C’est un langage de programmation libre qui est impératif (exécute une suite d’instructions élémentaires) orienté objet (organisation du code en objet). Il s’exécute côté serveur en PHP pour rendre côté client du HTML.  Il représente 82% des parts de marché sur le Web en 2016. En 2018, c’est près de 80% des sites web qui utilisent le langage PHP.

La dernière version de PHP, c’est la version 8 . Ce langage de script utilisé côté serveur à l’avantage d’être facile d’accès car il est peu typé, même si aux fils des versions cela ce vérifie de moins en moins, surtout depuis la version 7.

Il a aussi l’avantage de communiquer avec la base de données, ce qui en fait le lien parfait entre le côté serveur et le côté client. Par exemple, en récupérant les données d’un formulaire avec PHP, il est possible de les enregistrer en base de données.

# Les conventions de nommage

Pour le écrire une variable, une fonction ou un objet, il existe deux convention de nommage.

Il est conseillé de suivre une de ces conventions ou les deux pour se repérer plus facilement dans le code.

La convention « camelCase » qui permet d'écrire le nom de la variable, comme ceci: « leNomDeMaVariable ».

Vous remarquerez que la première lettre n'est pas en majuscule, c'est seulement quand un autre mot est employé à la suite que l'on rajoute une majuscule pour le démarquer.

Tandis que la convention « PascalCase » reprends la même démarche mais en ajoutant une majuscule sur le premier mot.

Des deux conventions, c'est le camelCase qui est le plus utilisé !

```
$maMagnifiqueVariable = '';

$MaMagnifiqueVariable = '';
```

# PHP

Déclaration PHP

# Balises de PHP

```php
<?php
    $bonjour = 'Bonjour tout le monde !';
    $text = 'Lorem ipsum dolor sit amet, consectetur adipisicing elit. Aperiam asperiores assumenda doloremque, eius illo, iste libero modi nisi, obcaecati quas quia sequi? Adipisci alias eaque fugiat quaerat quisquam voluptas, voluptatibus.';
?>
<!doctype html>
<html lang="fr">
    <head>
        <meta charset="utf-8">
        <title>titre de la page</title>
    </head>
    <body>
        <h1><?php echo $bonjour ?></h1>
        <p><?= $text ?></p>
    </body>
</html>
```

Un fichier PHP doit contenir l'extension « .php » pour fonctionner comme tel. Il est donc possible de remplacer l'extension d'un fichier HTML par celle de PHP pour inclure du code de celui-ci dans le code HTML.

Evidement pour que l'interprétation du code PHP se fasse, il faut implémenter les balises ouvrantes et fermantes de ce langage: « <?php » et « ?> » (respectivement la balise ouvrante et la balise fermante).

Comme vous pouvez le voir sur l'exemple droite, le code interprété par PHP se trouve entre les deux balises. Petite exception pour la fonction « echo » qui utilise « <?=» comme raccourci de balise ouvrante.

# PHP

Les commentaires

# Les commentaires

```php
<?php
    //mes variables
    $bonjour = 'Bonjour tout le monde !';
    /**
     * Pour des commentaires sur plusieurs lignes
     */
?>
```

Comme vous pouvez le voir dans cet exemple, il est possible de commenter de 2 façons différentes.

La première, c'est sur une ligne avec « // ». Très pratique pour ce noter un petit rappel.

La seconde, c'est sur plusieurs lignes avec « /** » pour démarrer les commentaires. Ensuite viens le « * » à chaque nouvelle ligne de commentaires. Puis le « */ » pour terminer les commentaires.

# PHP

Les variables

# La declaration des variables

```php
<?php
    $bonjour = 'Bonjour tout le monde !';
    $text = 'Lorem ipsum dolor sit amet, consectetur adipisicing elit. Aperiam asperiores assumenda doloremque, eius illo, iste libero modi nisi, obcaecati quas quia sequi? Adipisci alias eaque fugiat quaerat quisquam voluptas, voluptatibus.';
?>
```

En PHP, chaque variable commence par un « $ ».

C'est le signe distinctif pour déclarer une variable.

Contrairement à JavaScript, il faut continuer à mettre le « $ » a chaque appel d'une variable.

Dans cet exemple, vous avez deux variables qui contiennent des données de type texte (ou string en Anglais).

# PHP

Les opérateurs

# Les opérateurs logique

```php
if(!empty($maMagnifiqueVariable) || $valid === true && $text !== null){
    //execute code si rentre dans la condition
}
```

Il existe plusieurs types d'opérateurs.

Les opérateurs dit « logique » :
le « || » pour représenter le « ou »
et le « && » pour représenter le « et »
ils amènent une toute autre complexité sur une condition.

# PHP

Opérateurs de base

# Les opérateurs de base

```php
$five = 5;
$number = 254;

if($five >= $number){
    //execute code si rentre dans la condition
}
```

Les opérateurs dit « de comparaison » permettent de comparer

si une variable est égale à avec « === »

si elle est différente de avec « !== »

Il est aussi possible de comparer des chiffres avec « < », « = » ou « > ».

# PHP

Opérateurs d’incrémentation / décrémentation

# Incrémenter / décrémenter

```
$five = 5;
$number = 254;
```

```
$five++;
$five = $five + 1;
$number--;
$number = $number - 1;
```

Pour ajouter 1 ou  enlever 1 sur un nombre, il suffit de rendre égale cette même variable mais en lui ajoutant ou en lui enlevant 1.

Pour aller vers plus d'efficacité, il préférable d'utiliser l'opérateur « -- » qui enlèvera 1

tandis que l'opérateur « ++ » ajoutera 1.

# PHP

Opérateurs de concaténation

# Concaténer des chaînes de caractères

```php
$lorem ='Lorem ipsum dolor sit amet, consectetur adipisicing elit.';

$lorem2 =' A accusantium ad aliquid aperiam asperiores beatae ea earum est ex hic ipsum natus, nulla porro, quae quia tempora, velit veritatis vitae.';

$fullText = $lorem . $lorem2;
```

```php
<?= $fullText ?>
<?php echo $fullText ?>
```

Pour enchaîner deux chaînes de caractères ensembles, il faut les « concaténer ».

La variable « fullText » contient la concaténation de deux chaînes de caractères qui sont respectivement dans les variables « lorem » et « lorem2 ».

Ensuite il suffit d'afficher les variables dans le HTML soit en utilisant la fonction « echo » soit avec le raccourci « = ».

# PHP

Opérateurs de comparaison

# Les différents opérateurs de comparaison

| | | |
|---|---|---|
| $a == $b | Egal | **true** si *$a* est égal à *$b* après le transtypage. |
| $a === $b | Identique | **true** si *$a* est égal à *$b* et qu'ils sont de même type. |
| $a != $b | Différent | **true** si *$a* est différent de *$b* après le transtypage. |
| $a <> $b | Différent | **true** si *$a* est différent de *$b* après le transtypage. |
| $a !== $b | Différent | **true** si *$a* est différent de *$b* ou bien s'ils ne sont pas du même type. |
| $a < $b | Plus petit que | **true** si *$a* est strictement plus petit que *$b*. |
| $a > $b | Plus grand | **true** si *$a* est strictement plus grand que *$b*. |
| $a <= $b | Inférieur ou égal | **true** si *$a* est plus petit ou égal à *$b*. |
| $a >= $b | Supérieur ou égal | |

# PHP

Opérateurs logiques

# Les différents opérateurs logique

| | | |
|---|---|---|
| $a and $b | And (Et) | **true** si *$a* ET *$b* valent **true**. |
| $a or $b | Or (Ou) | **true** si *$a* OU *$b* valent **true**. |
| $a xor $b | XOR | **true** si *$a* OU *$b* est **true**, mais pas les deux en même temps. |
| ! $a | Not (Non) | **true** si *$a* n'est pas **true**. |
| $a && $b | And (Et) | **true** si *$a* ET *$b* sont **true**. |
| $a \|\| $b | Or (Ou) | **true** si *$a* OU *$b* est **true**. |

Cependant attention à l'utilisation des opérateurs « and », « or » car ils n'ont pas la même façon de comparer les variables et n'ont pas les mêmes priorités.

Il est préférable de privilégier « ! », « && » et « || » pour faire des comparaisons plus strictes.

# PHP

Les conditions (IF/SWITCH)

# Le “IF”

```php
$i = 5;

if ($i === 0) {
    echo "i égal à 0";
} elseif ($i === 1) {
    echo "i égal à 1";
} elseif ($i === 2) {
    echo "i égal à 2";
} elseif ($i === 3) {
    echo "i égal à 3";
} elseif ($i === 4) {
    echo "i égal à 4";
} elseif ($i === 5) {
    echo "i égal à 5";
} else {
    echo "i est supérieur à 5";
}
```

Le « if » permet d’émettre une condition.

Si i === 0 alors le code entre les accolades s’exécute sinon il passe à la condition suivante avec un « elseif ».

Et pour finir, si la variable i ne rempli aucunes conditions, le code qui est entre les accolades du « else » (sinon an Anglais) sera exécuté.

# Le “Switch”

D’une façon différente mais dans un but commun le « switch » permet de passer une variable dans plusieurs conditions.

Le code « switch » entre plusieurs « case » pour déterminer si la variable i correspond à 0 puis 1 puis 2.

Si ce n’est pas le cas, cela veut dire que la variable ne correspond à aucuns des trois chiffres.

Le code qui s’exécutera sera celui qui se trouve dans « default ».

```php
$i = 5;

switch ($i) {
    case 0:
        echo "i égal à 0";
        break;
    case 1:
        echo "i égal à 1";
        break;
    case 2:
        echo "i égal à 2";
        break;
    default:
        echo 'Merci de faire un choix...';
        break;
}
```

# PHP

Les boucles (WHILE/FOR/FOREACH)

# Les boucles avec “while”

```php
$i = 1;
while ($i <= 10) {
    echo $i++;  /* La valeur affichée est $i
avant l'incrémentation
                (post-incrémentation)  */
}

$i = 1;
while ($i <= 10):
    echo $i;
    $i++;
endwhile;
```

Le « while » peut se déclarer de deux façons différentes comme sur l’exemple de gauche.

Ce type de boucle se traduirait par « tant que » i n’est pas inférieur ou égal à 10, le code qui est dans le « while » continuera à s’exécuter.

Il faut bien sur penser à incrémenter i à chaque boucle pour éviter qu’il reste sur la même valeur et crée une boucle infinie.

# Les boucles avec “for”

Contrairement au « while », la boucle « for » doit être délimitée.

Elle comprends trois paramètres.

Le premier s’est pour initialiser la variable i avec un chiffre qui définira le démarrage de la boucle.

Ensuite, le deuxième paramètre permets de mettre une limite au nombre de boucles.

Et enfin, le troisième incrémente i a chaque itération de la boucle.

```php
for ($i = 1; $i <= 10; $i++) {
    echo $i;
}
```

# Les boucles avec “foreach”

```php
$arr = array(1, 2, 3, 4);
foreach ($arr as $value) {
    $value = $value * 2;
}

foreach ($arr as $key => $value) {
    $value = $value * 2;
}
```

Le « foreach » sert uniquement à boucler dans un tableau.

On y passe en paramètre le tableau dans lequel il faut boucler suivi de « as » puis d’une variable qui prendra comme valeur la partie du tableau qui sera analysé dans la boucle.

Il est aussi possible de rajouter un paramètre qui prendra en compte l’index du tableau avec « key » suivi de « => » puis de la valeur.

Pratique pour jouer sur les index d’un tableau.

# PHP

Les tableaux (numérotés / associatifs)

# Les tableaux en PHP

```php
$fruits2 = array("orange", "banane", "pomme");

$fruits3 = ["orange", "banane", "pomme"];
```

Comment déclarer un tableau en PHP ?

En utilisant la fonction « array » suivi de parenthèses. Les valeurs du tableau sont mises entre les parenthèses.

Ou plus simplement en utilisant les « [] » et en mettant les valeurs entre ceux-ci.

C'est cette deuxième manière de construire un tableau qui est désormais privilégiée.

# Les tableaux numérotés

Dans un tableau, les index permettent de ranger ou de trier les données plus facilement.

Les tableaux numérotés ont leur index sous forme de chiffres comme le tableau qui se trouve dans la variable « fruits1 ».

```
$fruits1 = array(
1 => "orange",
2 => "banane",
3 => "pomme«
);
```

# Les tableaux associatifs

```
$fruits = array(
"a" => "orange",
"b" => "banane",
"c" => "pomme«
);
```

Dans ce tableau, les index sont des lettres.

C'est ce que l'on appelle un tableau associatif.

On peut aussi y mettre des noms ou des chaînes de caractères plus longues.

# PHP

Les fonctions de PHP

# Qu’est-ce qu’une fonction ?

```php
function returnString($arg_1, $arg_2)
{
    $string = $arg_1 . $arg_2;

    return $string;
}
```

```php
$fullText = returnString($lorem, $lorem2);
```

Une fonction c’est un morceau de code qui est réutilisable à différents endroits dans le code.

Le mot clé « function » permet de déclarer celle-ci.

On mets les paramètres entre les parenthèses

et le code entre les accolades.

En PHP, il existe principalement deux types de fonctions, celles interne à PHP (cf. https://www.php.net/manual/fr/funcref.php ) comme par exemple: « echo », « substr », « gettype », etc...

Et celles que le développeur mets en place lui-même (des fonctions personnalisées).

# Les fonctions anonymes

Les fonctions anonymes sont tout simplement des fonctions sans nom que le développeur mets en place lui-même.

Elles peuvent être utilisé pour créer des fonctions courtes ou de rappel.

```php
// Pas de "use"
$example = function ($message) {
    var_dump($message);
};
$example($message);

// Hérite $message
$example = function () use ($message) {
    var_dump($message);
};
$example();
```

# PHP

Créer ses propres fonctions

# Un exemple de fonction

```php
function sumOfMinimum($numbersArray) {
    $minimums = [];

    foreach ($numbersArray as $item){
        $minimums[] .= min($item);
    }

    $sumOfMinimum = array_sum($minimums);

//    echo '<pre>';
//    var_dump($minimums);
//    echo '</pre>';

    //return sum of minimum find in arrays
    return $sumOfMinimum;
}

sumOfMinimum([[7, 9, 8, 6, 2], [6, 3, 5, 4, 3], [5, 8, 7, 4, 5]])
```

Voici un exemple de fonction personnalisée.

Dans cette fonction, c'est un tableau en deux dimensions qui est passé en paramètre.

Ensuite, la fonction est chargée de trouver les nombres les plus petits pour ensuite les additionner et retourner la somme de ces nombres.

# PHP

La transmission de données

# La session

```
session_start();
```

```
$_SESSION['mail'] = $mail;
$_SESSION['firstname'] = $result['firstname'];
$_SESSION['lastname'] = $result['lastname'];
```

En PHP, il est possible de transmettre les données entre les pages de différentes manières.

Les « superglobales » sont là pour répondre à ce besoin. Ce sont des variables internes à PHP qui sont toujours disponibles quelque soit le contexte ce qui permet de les utilisées d'une page à une autre.

En voici quelques unes: *$_SERVER*, *$_GET*, *$_POST*, *$_FILES*, *$_COOKIE*, *$_SESSION*, *$_REQUEST*, *$_ENV*

La superglobale « $_SESSION » permet de stocker des données tout au long de la session de l'utilisateur si elle est démarrée.

# Les cookies

Pourquoi utiliser les cookies ? Pour stocker les préférences de l'utilisateur, les informations permettant de l'identifier à nouveau, etc...

**La première possibilité** de manipuler les cookies, c'est grâce à la super globale « $_COOKIE ».

Cette variable renvoi un tableau de tous les cookies à l'instant T. Si un cookie se crée lors de l'ouverture de la page, il faut attendre le rechargement de celle-ci ou le passage à une autre page pour voir le cookie en question.

**La deuxième possibilité**, c'est d'utiliser la fonction « setcookie() ». Elle permet de créer un cookie en lui passant en paramètres le nom du cookie, les données à enregistrer dedans et le temps de validité du cookie. Pour supprimer un cookie, il suffit juste de mettre le nom en paramètre. Le fait de ne pas mettre la date de validité supprime le cookie directement.

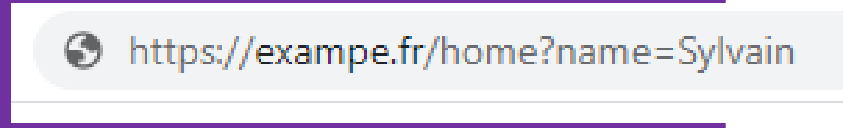
```php
echo var_dump($_COOKIE);
echo var_dump($_COOKIE['PHPSESSID']);
//    crée des cookies
   setcookie('test', 'Bonjour à tous !', (time() + 3600), 'cookies-storage');
   setcookie('essai[miam]', 'c\'est bon', (time() + 3600)); //expire dans une heure timestamp unix 1 janvier 1970
   $value ='Lorem ipsum dolor sit amet, consectetur adipisicing elit. Corporis culpa cum cumque debitis excepturi repellat saepe suscipit ut veritatis voluptates? Alias aut beatae blanditiis, enim expedita nisi reprehenderit similique ullam.';
   setcookie( "TestCookie", $value, strtotime( '+30 days' ) ); //expire dans trente jours
   setcookie( "TestCookies", $value, time()+(60*60*24*30) ); //expire dans trente jours
   setcookie('essai[génial]', 'c\'est génial !', (time() + 3600));
   setcookie('essai[génialOne]', 'c\'est génial !', (time() + 3600));
   setcookie('essai[génialTwo]', 'c\'est génial !', (time() + 3600));
   setcookie('essai[génialThree]', 'c\'est génial !', (time() + 3600));
//    efface un cookie
   setcookie('essai[génial]'); //juste en enlevant la durée, cela supprime le cookie
//    affiche un cookie
//    echo $_COOKIE['test'];
   print_r($_COOKIE['test']['miam']);

   if (isset($_COOKIE['essai'])) {
      foreach ($_COOKIE['essai'] as $name => $value) {
         $name = htmlspecialchars($name);
         $value = htmlspecialchars($value);
         echo "$name : $value <br />\n";
      }
```

# PHP

La transmission via url avec la méthode GET

# Transmettre des données via l'url




```php
echo 'Bonjour ' . htmlspecialchars($_GET["name"]) . '!';
```

Les données peuvent être transmises via l'url grâce à la super globale « $_GET ».

Dans l'url d'exemple, un paramètre est rajouté avec ?nomDuParametre=Valeur

C'est ce paramètre qui vas être récupéré avec le « name » qui se trouve dans le tableau de la super globale.

# PHP

La transmission via formulaire avec la méthode POST

# Transmettre des données via un formulaire

```
<form method="post" class="mb-5">
    <div class="row mb-5">
        <div class="col-xs-12 col-md-3">
            <label for="mail" class="text-white">Email:</label>
        </div>
        <div class="col-xs-12 col-md-9">
            <input type="email" class="form-control" id="mail" name="mail"
placeholder="Entrez votre email" required>
            <div id="erreur-mail"></div>
        </div>
    </div>
    <div class="row mb-5">
        <div class="col-xs-12 col-md-3">
            <label for="password" class="text-white">Mot de passe:</label>
        </div>
        <div class="col-xs-12 col-md-9">
            <input type="password" class="form-control" id="password"
name="password" placeholder="Entrez votre mot de passe" required>
            <div id="erreur-password"></div>
        </div>
    </div>
    <div class="col-12 text-center">
        <button id="envoi" type="submit" class="btn btn-color">Se
connecter</button>
    </div>
</form>
if (isset($_POST['mail'], $_POST['password'])) {
    $mail =
htmlspecialchars(trim($_POST['mail']));
    $password =
```

Les données d'un formulaire peuvent être récupérer avec « $_GET », « $_POST » ou « $_FILES ».

Généralement, ce sont les deux dernières qui sont utilisées.

« $_FILES » est pratique pour récupérer les fichiers uploadés par l'utilisateur.

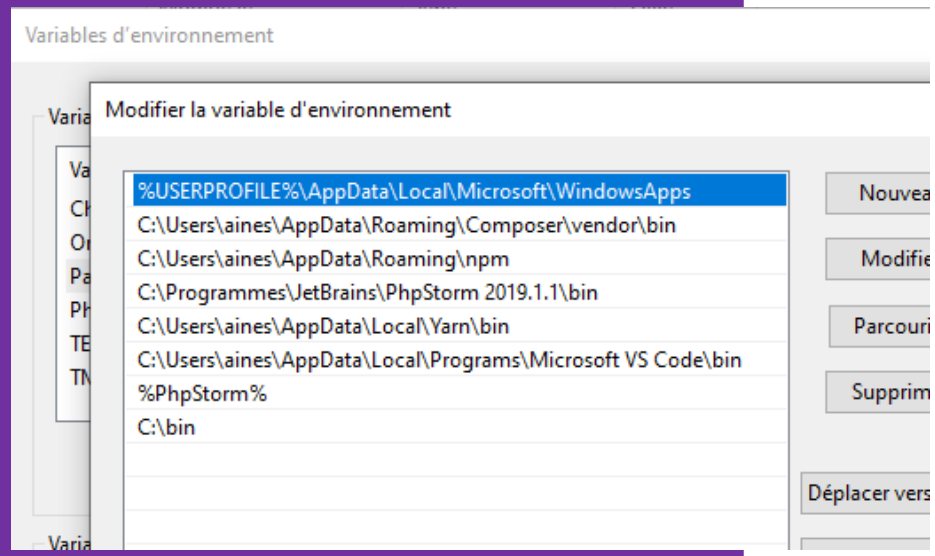
« $_POST » permet de récupérer les différents champs d'un formulaire.

Il suffit d'ajouter « post » dans l'attribut « method » de la balise « form » pour ensuite pouvoir récupérer les données soumisses par l'utilisateur grâce aux données qui se trouvent dans « $_POST ».

# PHP

Préparation de l’espace de développement

# Installer PHP sur votre système




- Avant d'installer PHP, sachez que si vous installer XAMPP, vous n'aurez pas besoin de faire cette installation, ce logiciel fait tout pour vous. Sinon, si vous avez besoin d'installer une version de PHP en global sur votre machine pour une utilisation plus poussé du langage, n'hésitez pas à faire ce qui vas suivre.

- Pour installer la dernière version de PHP, il faut se rendre sur le site : https://www.php.net/downloads.php

- Au passage, le site php.net propose toute la documentation nécessaire au langage PHP. Vous téléchargez la dernière version stable selon l' OS (operating system, par exemple Windows, Linux ou MacOS) qui se trouve sur votre machine.

Une fois télécharger, il faut le décompresser et le coller sur le disque dur qui acceuille votre système.

Ensuite il faut aller dans les variables d'environnements de votre système. Sur Windows, tapez « path » dans la barre de recherches ajouter une variable à l'environnement en spécifiant le chemin d'accès à PHP.

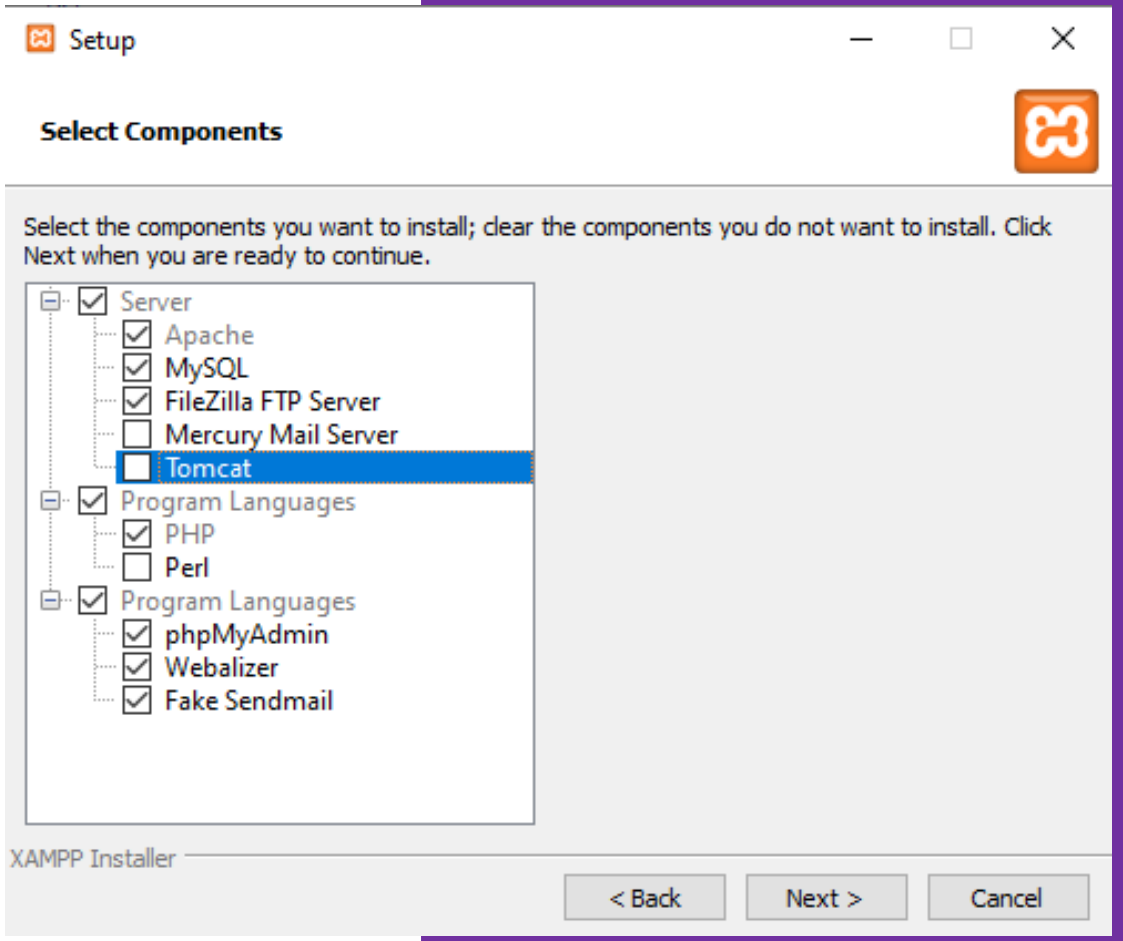
# Installer XAMPP

1. **Avant d'installer XAMPP**, il suffit de se rendre sur la page: https://www.apachefriends.org/fr/download.html

Pour télécharger XAMPP avec la version 7.3 de PHP. Et vérifier que cet installeur correspond à l'architecture (x82 ou x64) de votre machine (Windows, Mac ou Linux).

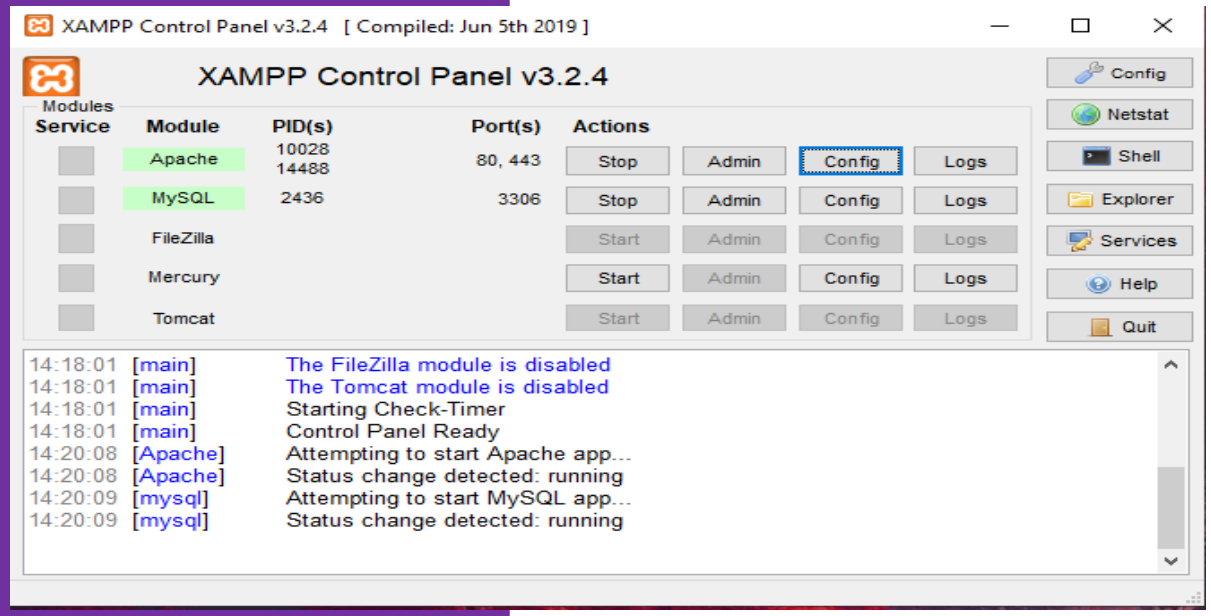
2. **Lancez l'installeur de XAMPP** pour pouvoir configurer les outils et les langages à installer sur votre machine.

Comme sur la capture de droite, n'hésitez pas à décocher ce qui ne sera pas utile avec PHP. Ensuite vous n'avez plus qu'a laisser le champ prérempli ou tout simplement mettre autre une destination pour l'enregistrement des fichiers.



# Configuration Apache

Qu'est-ce qu'Apache ? C'est un serveur HTTP (il répond aux requêtes du World Wide Web en utilisant le protocole HTTP en gérant le dialogue entre le serveur et le client). De base, que ce soit XAMPP ou WAMP, la configuration d'Apache est déjà suffisante. Mais selon vos besoins, vous pouvez configurer les variables d'environnement,  les hôtes, PHP, les extensions, etc…




1 Vous pouvez configurer Apache en cliquant sur le bouton « config » sur la ligne correspondante. Puis sélectionnez « apache (httpd.conf) » pour ouvrir le fichier de configuration d'Apache.

2 Le fichier de configuration d'Apache est un fichier texte qui regroupe toute la configuration. Si vous cherchez quelque chose en particulier, je vous conseille de faire un « ctrl+r » pour trouver plus rapidement le fichier en question.

Il est aussi possible de modifier des configurations misent en commentaires en enlevant le « # » en début de ligne. Ceci aura pour effet d'activer la configuration concernée. De plus vous pouvez modifier des valeurs comme les adresses, les tailles allouées à la mémoire, créer des « hosts », etc…