

# Programmation avancée & Symfony

BUT Informatique - 3ème année

## TP 1

### Install. & config.

Claire BOMBEAUX  
Version : **06/09/2025**



#### Objectifs du TP 1

- Installer et configurer un serveur web local (*WampServer*)
  - Installer *Composer*
  - Installer *Symfony*, le configurer & créer votre projet
  - Activer le *Profiler* de *Symfony*
  - Activer le *mode DEBUG* de *Symfony*
-

# Sommaire

■ Objectifs du TP 1 .....	0
■ Sommaire.....	1
1. Installation de WampServer et Composer.....	2
1.1 WampServer .....	2
Intro : Qu'est-ce que WampServer ? .....	2
Installer WampServer .....	3
1.2 Composer .....	7
Installer Composer.....	7
Ajouter PHP au PATH .....	10
Vérifier l'installation de Composer .....	10
1.3 Configuration de PHP (via WampServer).....	11
2. Création et config. d'un projet Symfony.....	13
2.1 Intro. : Versions et prérequis .....	13
Versions ( <i>Releases</i> ) de Symfony .....	13
Prérequis techniques ( <i>requierements</i> ) .....	13
2.2 Création du projet Symfony .....	14
2.3 Activer le Profiler de Symfony.....	16
Installation et configuration du Profiler .....	16
2.4 Configuration .....	18
Alias, Virtuals Hosts & Fichiers .htaccess Apache.....	18
Environnements Symfony.....	22
Activer le mode DEBUG .....	22
Conclusion .....	23
[TP 1] Annexe 1 : Commandes utiles.....	24
Installer/créer un projet Symfony .....	24
Voir les commandes disponibles .....	24
Connaître sa version de Symfony .....	24
Mettre à jour composer, les dépendances.....	24
Vider le cache.....	24
Debug config .....	24
Obtenir de l'aide sur une commande .....	24

# 1. Installation de WampServer et Composer

## 1.1 WampServer

Intro : Qu'est-ce que WampServer ?



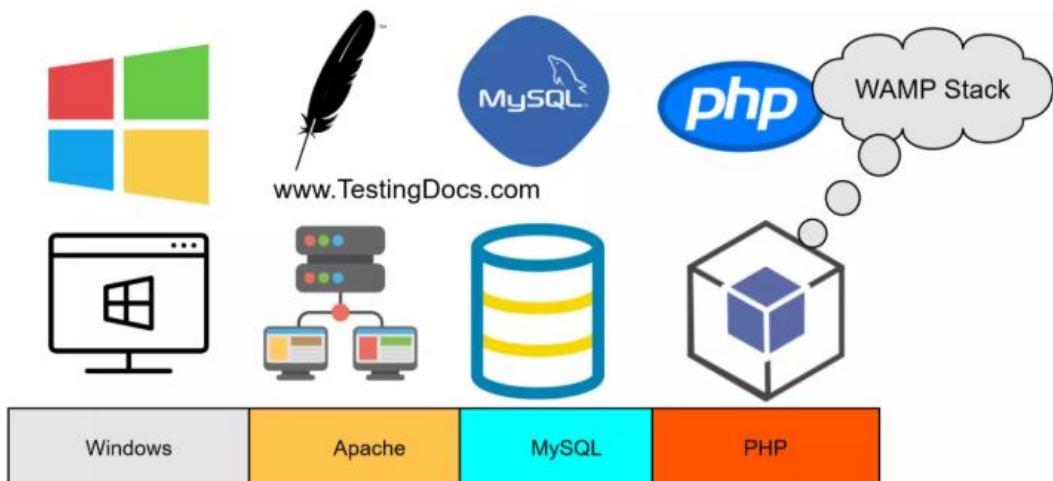
WampServer est un environnement de développement web local pour Windows qui intègre Apache (serveur web), MySQL (base de données), PHP et phpMyAdmin dans un package facile à installer et configurer.

WampServer permet de créer un **serveur web local** sur votre machine **Windows**, vous donnant accès à :

- **Apache** : serveur web (HTTP) pour servir vos pages

Serveur frontal. est « devant » tous les autres et répond directement aux requêtes du client web (navigateur) ;

- **MySQL** : système de gestion de base de données (SGBD). Stockage des données.
- **PHP** : langage de programmation côté serveur. Pages web dynamiques (générées à la demande. Variables)
- **phpMyAdmin** : interface web pour gérer vos bases de données MySQL

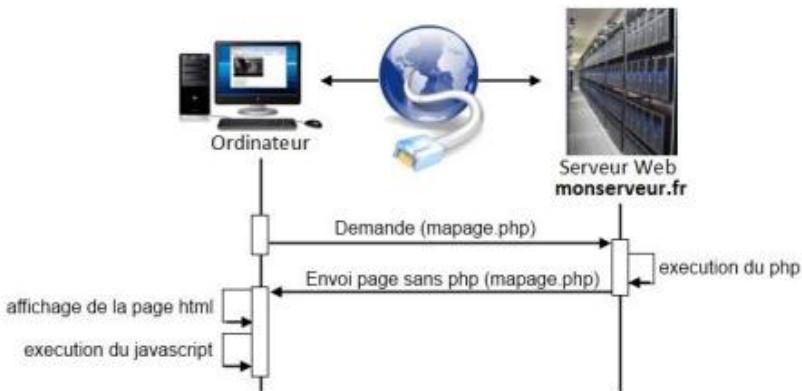


Src. : <https://www.testingdocs.com/testlink-install-requirements/>

Il existe LAMP pour « Linux, Apache, MySQL, PHP ».

Par défaut, les sources de vos projets sont à mettre dans le répertoire www de WampServer.

## Fonctionnement



Un navigateur seul ne peut pas lire le PHP, seul le serveur le peut. Il faudra donc déposer le site web dans un dossier spécial du serveur, le répertoire www ("www directory") accessible via Wamp.

Une fois le site web dans ce dossier, vous pourrez alors visualiser votre œuvre en vous connectant au "localhost". Celui-ci est accessible via le navigateur en tapant "localhost" dans la barre de navigation, mais également via WampServer en cliquant sur "localhost".

Src. : <https://webqburnet.com/wp-content/uploads/2021/05/8-Wampserver.pdf>

## Installer WampServer

Téléchargez WampServer depuis le site officiel : <https://www.wampserver.com>

Utiliser « passer au téléchargement direct » pour ne pas avoir à vous inscrire

Choisissez la version correspondant à votre système Windows (ex. : 64-bit) :  
[wampserver3.3.7 x64.exe](#) ...

 **Attention** Lire instructions utilisation.pdf + tenir compte des remarques et prérequis affichées lors de l'installation !

**— N'installez pas Wampserver PAR-DESSUS une version existante**, suivez les conseils de :

- Installer une nouvelle version de Wampserver : <http://forum.wampserver.com/read.php?1,119444>

**— Installez Wampserver dans un dossier à la racine d'un disque**, par exemple C:\wamp ou D:\wamp.

**Prenez un chemin d'installation qui ne comporte pas d'espaces ou de caractères diacritiques** ; donc, pas d'installation dans c:\Program Files\ ou c:\Program Files (x86)\

Il faut, **AVANT** l'installation, désactiver ou fermer certaines applications :

- **Fermer Skype** ou le forcer à ne pas utiliser le port 80

Item N°04 des CONSEILS DE DÉPANNAGE de Wampserver : <http://forum.wampserver.com/read.php?1,88043>

**- Désactiver IIS**

Item N°08 des CONSEILS DE DÉPANNAGE de Wampserver : <http://forum.wampserver.com/read.php?1,88043>

**Si vous pensez ne pas être en accord avec ces prérequis, validez Annuler pour annuler l'installation, appliquez les prérequis et relancez l'installation.**

**Ce programme nécessite des priviléges Administrateur pour fonctionner correctement. Il sera lancé avec l'option « Exécuter en tant qu'administrateur ». Si vous ne voulez pas qu'un programme ait cette option, annulez l'installation.**

**— Paquetages VC —**

Pour vérifier que tous les paquetages VC++ sont bien installés et avec les dernières versions, utilisez l'outil : [https://wampserver.aviaTechno.net/files/tools/check\\_vcRedist.exe](https://wampserver.aviaTechno.net/files/tools/check_vcRedist.exe) et vous trouverez tous les paquetages sur <https://wampserver.aviaTechno.net/> dans la section Visual C++ Redistributable Packages.

N'utilisez pas une ancienne version de l'outil que vous auriez déjà téléchargé. Téléchargez toujours la dernière version juste avant de l'utiliser.

Si vous avez un Windows en 64 bits, il faut installer les deux versions 32 et 64bits de chaque paquetage VC.

Pour plus de facilité, voir : Manière simple d'installer les Visual C++ Redistributable Packages

Cela inclut d'installer les packages VC++ nécessaires : <https://wampserver.aviaTechno.net> **avant** l'install. de WAMP.

#### Manière simple d'installer les Visual C++ Redistributable Packages

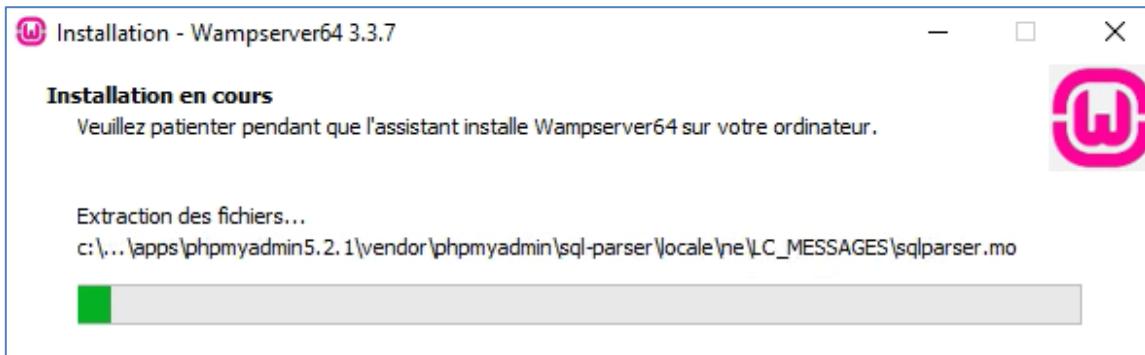
La manière la plus simple, la plus facile, la moins sujette à erreur, la plus rapide, d'installer les redistributables VC++ indispensables à Wampserver (et à plein d'autres logiciels) est d'utiliser un programme qui installe tout ce qui est nécessaire avec un seul exécutable.

Non, ce n'est pas une utopie ! Ça existe et c'est AIO Repack nommé **VisualCppRedist AIO** créé par *addobi1406*, toujours mis à jour avec les dernières versions de Microsoft.

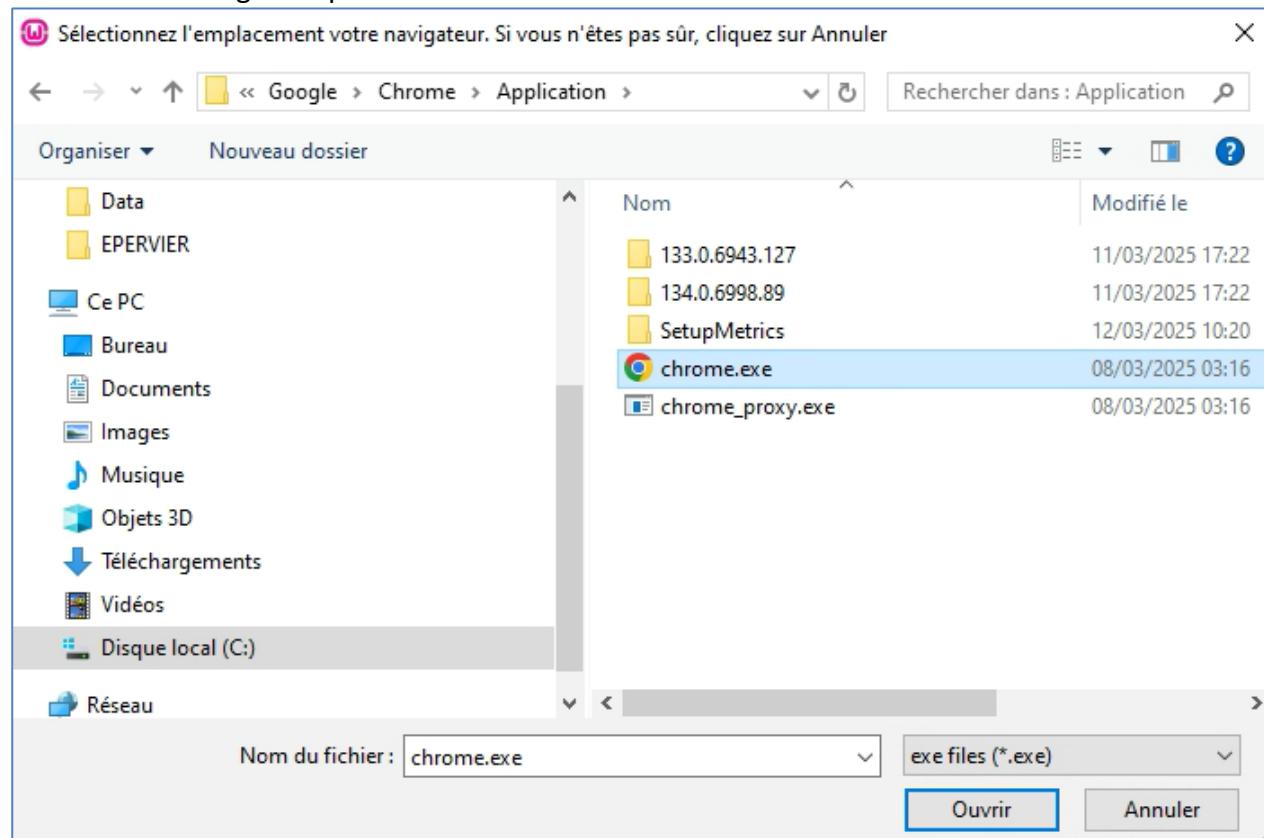
Les fichiers à télécharger sont là : [https://github.com/addobi1406/VisualCppRedist\\_AIO/releases](https://github.com/addobi1406/VisualCppRedist_AIO/releases) - La page Github est là : [https://github.com/addobi1406/VisualCppRedist\\_AIO](https://github.com/addobi1406/VisualCppRedist_AIO)

Télécharger la dernière version du fichier **VisualCppRedist\_AIO\_x86\_x64.exe** et le lancer en tant qu'administrateur, tous les VC++ redistributables nécessaires seront installés et les fichiers en trop ou obsolètes seront supprimés. Pour que tout se passe sans problème, il faudra désactiver certaines applications lorsque cela sera demandé par l'installateur des VC++ Redist.



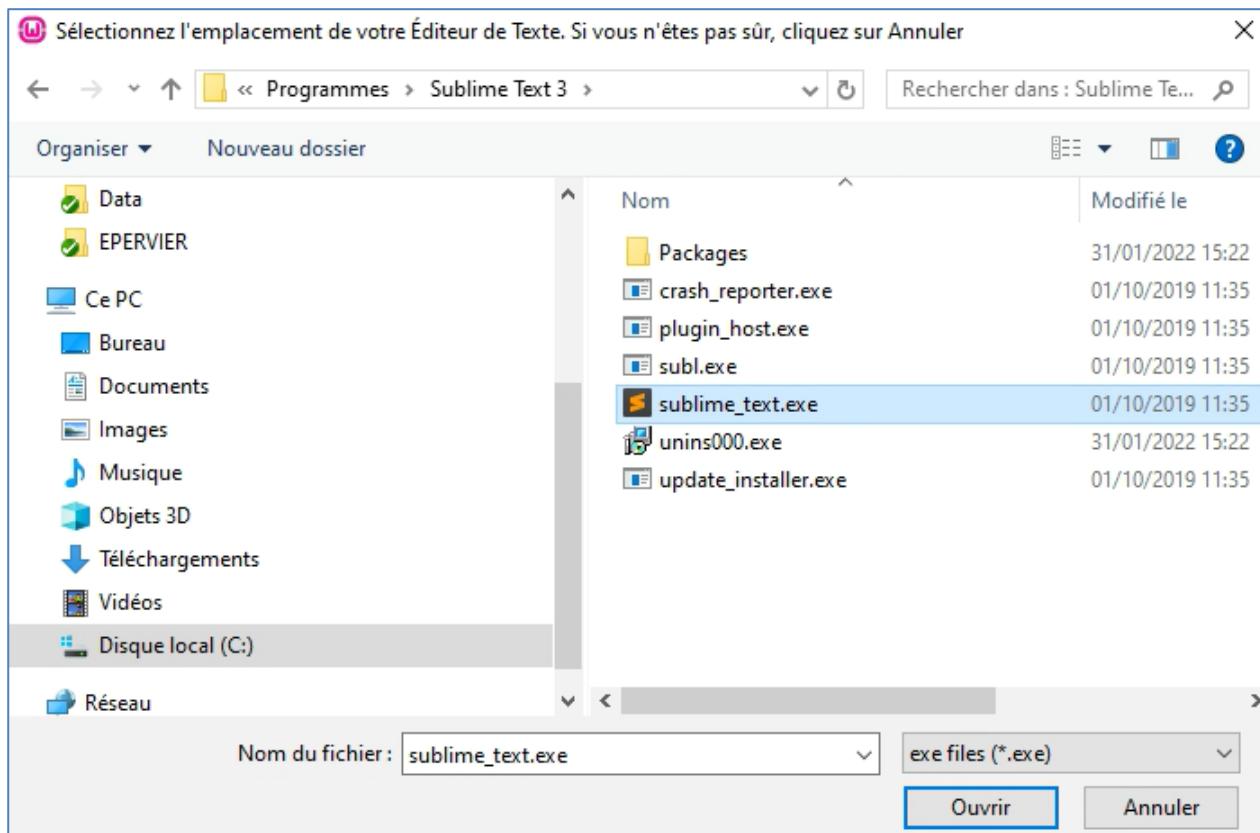


Choisir votre navigateur par défaut :



Choisir votre éditeur de texte par défaut (**Sublime Text** conseillé)

↗ Téléchargement Sublime Text : <https://www.sublimetext.com/download>



Démarrez WampServer



L'icône dans la barre des tâches doit être verte :

Rappel : Adresse localhost → 127.0.0.1

## 1.2 Composer

Composer est le gestionnaire de dépendances PHP indispensable pour Symfony.

En effet, il permet d'installer des centaines de packages librement disponibles. On les trouve sur Packagist . Composer permet de gérer les dépendances d'un projet et également de créer le squelette d'une application Symfony.

 Documentation : <https://getcomposer.org/doc/>

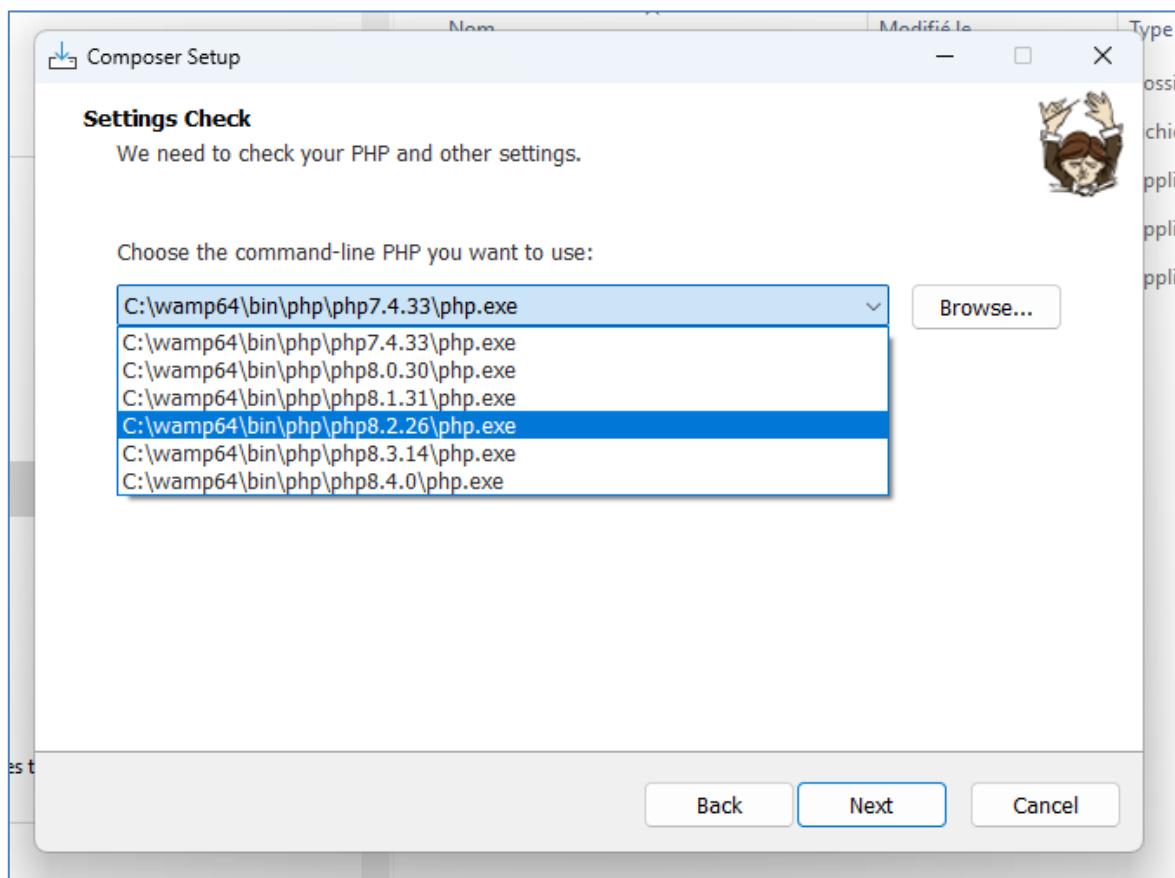
### Installer Composer

Plusieurs méthodes sont possibles, en voici quelques-unes :

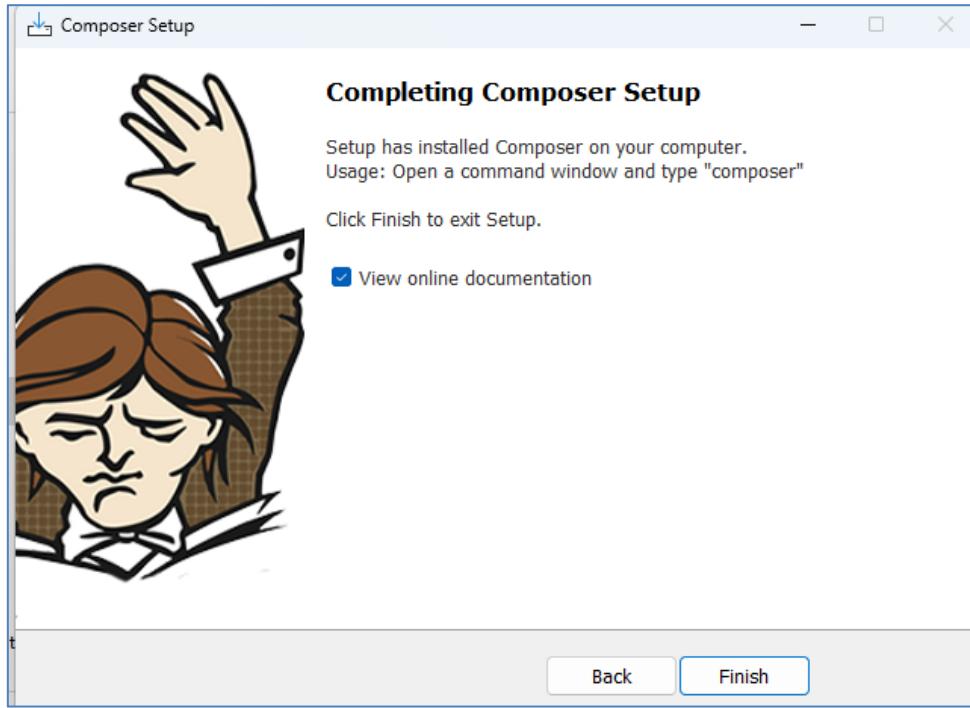
#### Méthode 1 : Installateur Windows (recommandé)

Téléchargez Composer-Setup.exe depuis le site officiel : [getcomposer.org](https://getcomposer.org)

Exécutez l'installateur et configurer Composer



Choisir la version de PHP utilisée (Pour cela voir la Partie 2. Création d'un projet Symfony > 2.1 Releases et Requierements de ce TP1).



Attention si Internet KO (type VM) : tenter de saisir votre proxy. Si aucun accès Internet, voir méthode manuelle

## Méthode 2 : Installation en ligne de cmdé

### Command-line installation

To quickly install Composer in the current directory, run the following script in your terminal. To automate the installation, use [the guide on installing Composer programmatically](#).

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
php -r "if (hash_file('sha384', 'composer-setup.php') === 'dac665fdc30fdd8ec78b38b9800061b4150413ff2e3b6f88543c'
php composer-setup.php
php -r "unlink('composer-setup.php');"
```

This installer script will simply check some `php.ini` settings, warn you if they are set incorrectly, and then download the latest `composer.phar` in the current directory. The 4 lines above will, in order:

- Download the installer to the current directory
- Verify the installer SHA-384, which you can also [cross-check here](#)
- Run the installer
- Remove the installer

Most likely, you want to put the `composer.phar` into a directory on your PATH, so you can simply call `composer` from any directory (*Global install*), using for example:

```
sudo mv composer.phar /usr/local/bin/composer
```

For details, [see the instructions on how to install Composer globally](#).

**WARNING:** Please do not redistribute the install code. It will change with every version of the installer. Instead, please link to this page or check [how to install Composer programmatically](#).

Src. : <https://getcomposer.org/download/>

### Détails des commandes :

- Chargement de l'installateur
- Vérification de l'intégrité (hash)
- Exécution de l'installateur
- Retire l'installateur

### **Méthode 3 : Installation hors ligne (pour information uniquement)**

Sur votre machine hôte (avec internet) :

Téléchargez le fichier `composer.phar` depuis [getcomposer.org](http://getcomposer.org)

#### **Manual Download**

If you prefer to download the phar manually, here are the available versions:

[Latest Stable \(sha256 / sha256sum / asc\)](#) for PHP 7.2+ users

[Latest Preview \(alpha/beta/RC\) \(sha256 / sha256sum / asc\)](#)

[Latest Snapshot \(sha256 / sha256sum\)](#)

[Latest 2.x \(sha256 / sha256sum / asc\)](#)

[Latest 2.2.x LTS \(sha256 / sha256sum / asc\)](#) for PHP 5.3 to 7.1. For PHP 7.2+ please use the latest version support policy for more information.

Téléchargez aussi `composer-setup.php` si nécessaire

Transférez ces fichiers vers votre VM (via Dossier partagé entre hôte et VM

ou Clé USB ou Réseau local si disponible)

Dans votre VM (sans Internet) :

Placez `composer.phar` dans le dossier PHP de WampServer :

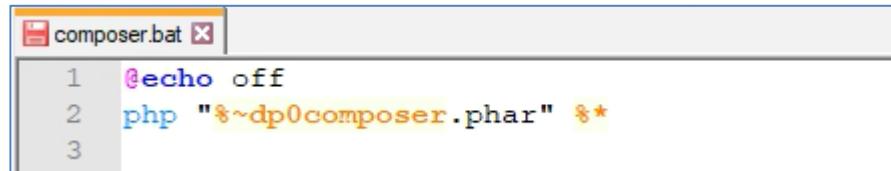
cmd

C:\wamp64\bin\php\php8.2.26\composer.phar

Créez un fichier batch `composer.bat` dans le même dossier :

batch

```
@echo off
php "%~dp0composer.phar" %*
```



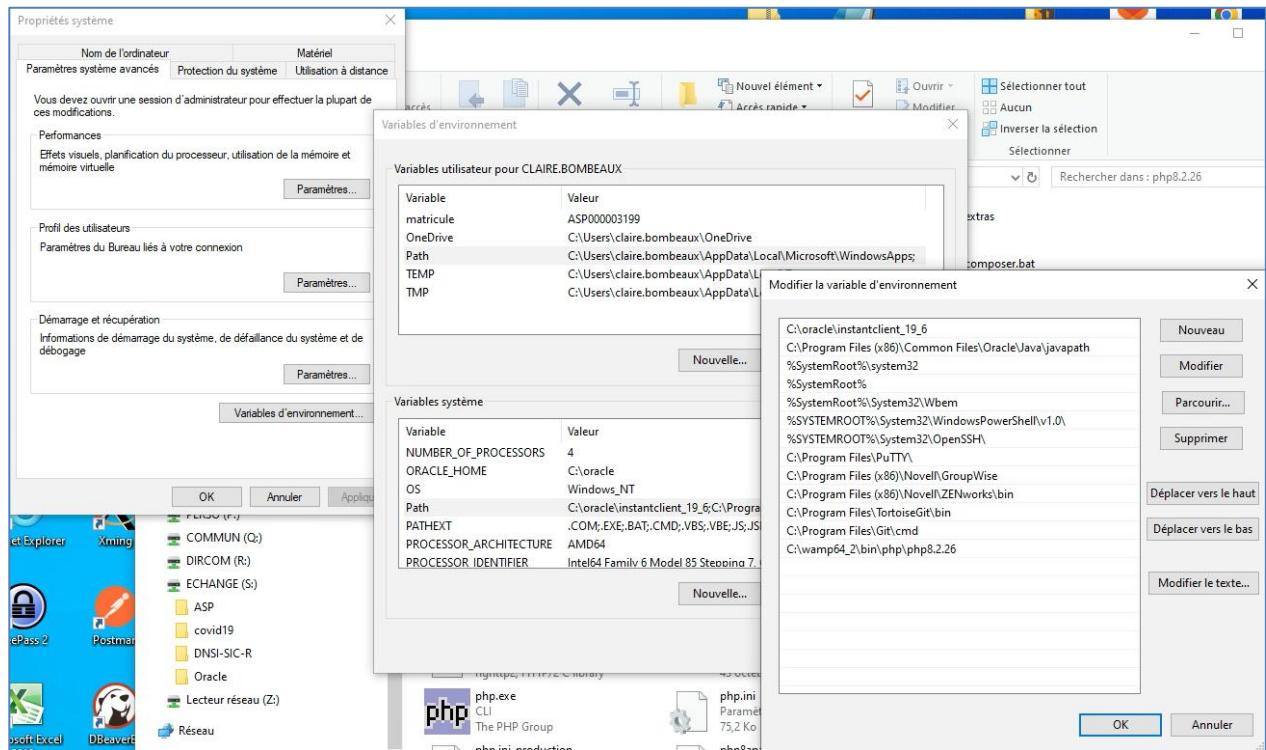
```
composer.bat
1 @echo off
2 php "%~dp0composer.phar" %*
```

Ajoutez ce dossier au PATH Windows de votre VM (Cf. partie suivante)

## Ajouter PHP au PATH

 **Requis** si non fait via l'install. de Composer

→ Variabes d'env. système



Ajouter le chemin de PHP (Ex. : `C:\wamp64\bin\php\php8.2.26`) au **PATH**

Redémarrer car les variables système se chargent au démarrage

`php -v`

*cmd*

**Vérifier l'installation de Composer**

`composer --version`

*cmd*

```
Windows PowerShell
PS C:\wamp64_2\bin\php\php8.2.26> composer --version
Composer version 2.8.9 2025-05-13 14:01:37
PHP version 8.2.26 (C:\wamp64_2\bin\php\php8.2.26\php.exe)
Run the "diagnose" command to get more detailed diagnostics output.
PS C:\wamp64_2\bin\php\php8.2.26>
```

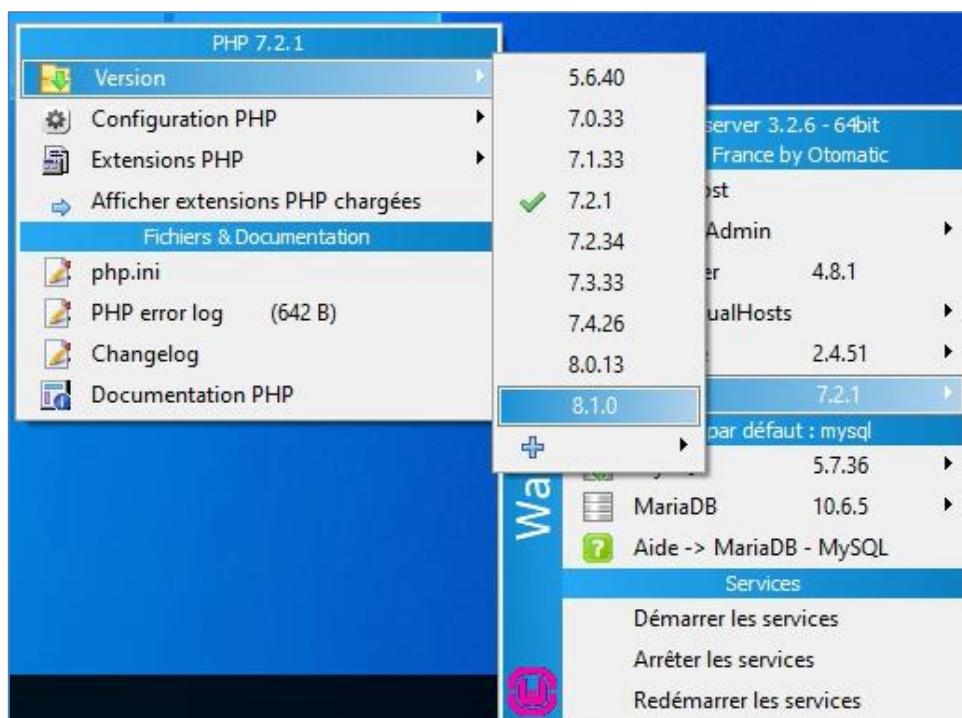
## 1.3 Configuration de PHP (via WampServer)

Après avoir installé WampServer, vous devez vous assurer que PHP est configuré correctement pour Symfony.

Pour cela, consultez les **requierements** de Sf : [Cf. Partie 2. Création d'un projet Symfony > 2.1 Releases et Requierements de ce cours.](#)

Vérifiez que PHP est en version **8.2** ou supérieure (requis pour Symfony **6+**)

Allez dans **PHP → Version** et sélectionnez **PHP 8.2** ou supérieur



Voir aussi partie précédente de ce TP : [Ajouter PHP au PATH \(cmd : php -v\)](#)

Vérifier / activez les extensions PHP suivantes, nécessaires à SF , dans WampServer :

- ✓ `php_curl`
- ✓ `php_intl`
- ✓ `php_mbstring`
- ✓ `php_openssl`
- ✓ `php_pdo_mysql`
- ✓ `php_xml`

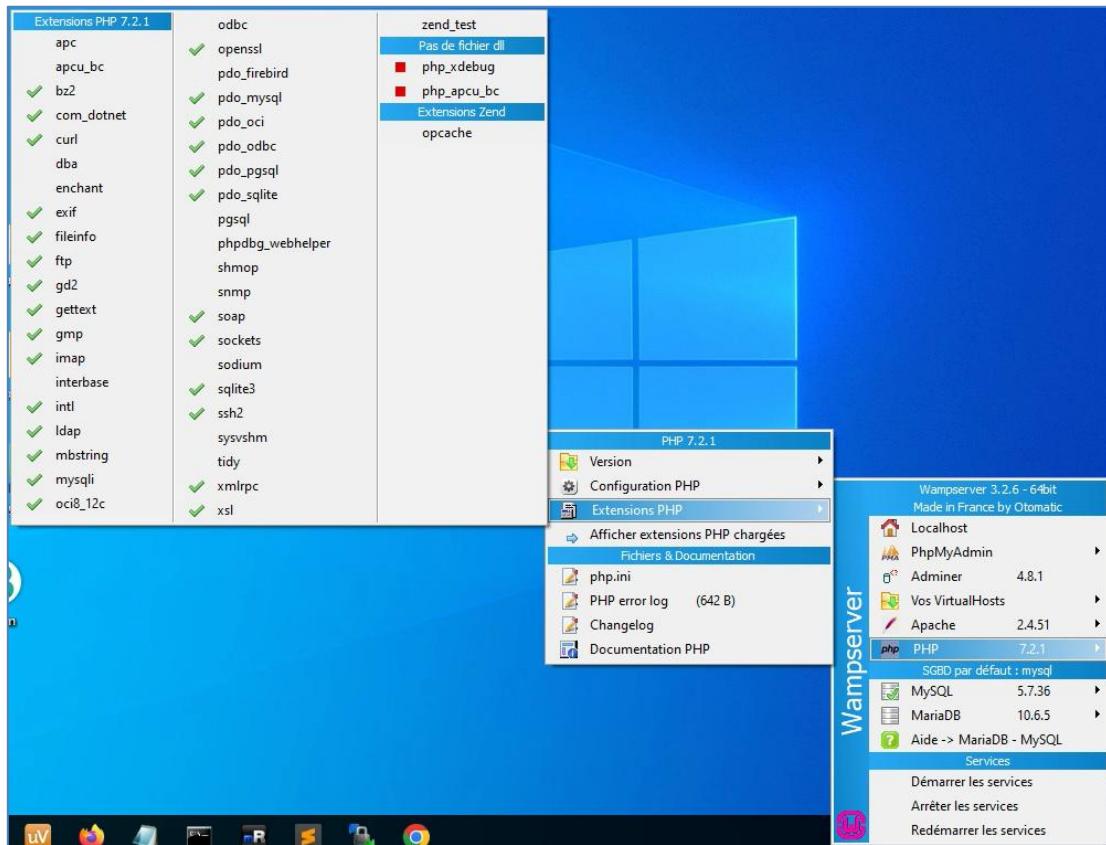
Redémarrer WampServer.

Une extension PHP est un **module complémentaire** qui ajoute des fonctionnalités supplémentaires au langage PHP de base. C'est comme un plugin qui étend les capacités du langage.

Activation d'une extension => fichier [php.ini](#)

A noter : il existe une fonctionnalité de WAMP permet de voir quelles extensions sont activées.

Clic gauche sur l'icône WampServer → PHP → Extensions PHP



Familiarisez-vous avec WAMP Server et ses fonctionnalités. Comprenez comment accéder aux fichiers de configurations et aux fichiers de logs.

## 2. Création et config. d'un projet Symfony

### 2.1 Intro. : Versions et prérequis

Avant toutes installations, il est généralement nécessaire d'identifier la version souhaitée et d'en connaître les prérequis logiciels et physiques.

#### Versions (Releases) de Symfony

<https://symfony.com/releases>

### Symfony Releases

Symfony releases follow a time-based model: minor versions come out every six months (in May and November); major versions come out every two years. Check out the [release process details](#).

Stable Release	Long-Term Support Release
7.3.1	6.4.23
<ul style="list-style-type: none"><li>Requires PHP 8.2.0 or higher</li><li>First released in May 2025</li><li><b>Recommended for most users</b></li><li>Includes the latest features</li><li>It's easier to upgrade to newer versions</li></ul>	<ul style="list-style-type: none"><li>Requires PHP 8.1.0 or higher</li><li>First released in November 2023</li><li>3 year support for bugs and security fixes</li><li>It doesn't include the latest features</li><li>It's harder to upgrade to newer versions</li></ul>

**Symfony 7.3** is backed by  Les-Tilleuls.coop

**Symfony 6.4** is backed by  PRIVATE PACKAGIST  SensioLabs

LTS : version stable d'un logiciel informatique, maintenue pendant une période plus longue que l'édition standard.

#### Prérequis techniques (requirements)

<https://symfony.com/doc/current/setup.html#technical-requirements>

### Technical Requirements

Before creating your first Symfony application you must:

- Install PHP 8.2 or higher and these PHP extensions (which are installed and enabled by default in most PHP 8 installations): [ctype](#), [iconv](#), [PCRE](#), [Session](#), [SimpleXML](#), and [Tokenizer](#);
- [Install Composer](#), which is used to install PHP packages.

Also, [install the Symfony CLI](#). This is optional, but it gives you a helpful binary called `symfony` that provides all tools you need to develop and run your Symfony application locally.

The `symfony` binary also provides a tool to check if your computer meets all requirements. Open your console terminal and run this command:

```
$ symfony check:requirements
```

## 2.2 Cration du projet Symfony

Ouvrez l'invite de commande Windows

Placez-vous dans le repertoire `www` de WampServer et creez votre projet :

*cmd*

`cd C:\wamp64\www`

`composer create-project symfony/skeleton:7.3.*-dev mon_projet`

```
C:\wamp64\www>composer create-project symfony/skeleton:"7.3.*-dev" mon_projet
Creating a "symfony/skeleton:7.3.*-dev" project at "./mon_projet"
Installing symfony/skeleton (7.3.x-dev c7e48b636a4139ca52fc050106a37fa0e64da56f)
  - Downloading symfony/skeleton (7.3.x-dev c7e48b6)
  - Installing symfony/skeleton (7.3.x-dev c7e48b6): Extracting archive
Created project in C:\wamp64\www\mon_projet
Loading composer repositories with package information
Updating dependencies
Lock file operations: 1 install, 0 updates, 0 removals
  - Locking symfony/flex (v2.8.1)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 1 install, 0 updates, 0 removals
  - Downloading symfony/flex (v2.8.1)
  - Installing symfony/flex (v2.8.1): Extracting archive
Generating autoload files
1 package you are using is looking for funding.
Use the `composer fund` command to find out more!

Run composer recipes at any time to see the status of your Symfony recipes.

Loading composer repositories with package information
Restricting packages listed in "symfony/symfony" to "7.3.*"
Updating dependencies
```

Note si proxy, le setter avant

```
set HTTP_PROXY=http://username:password@proxy.company.com:8080
set HTTPS_PROXY=https://username:password@proxy.company.com:8080
```

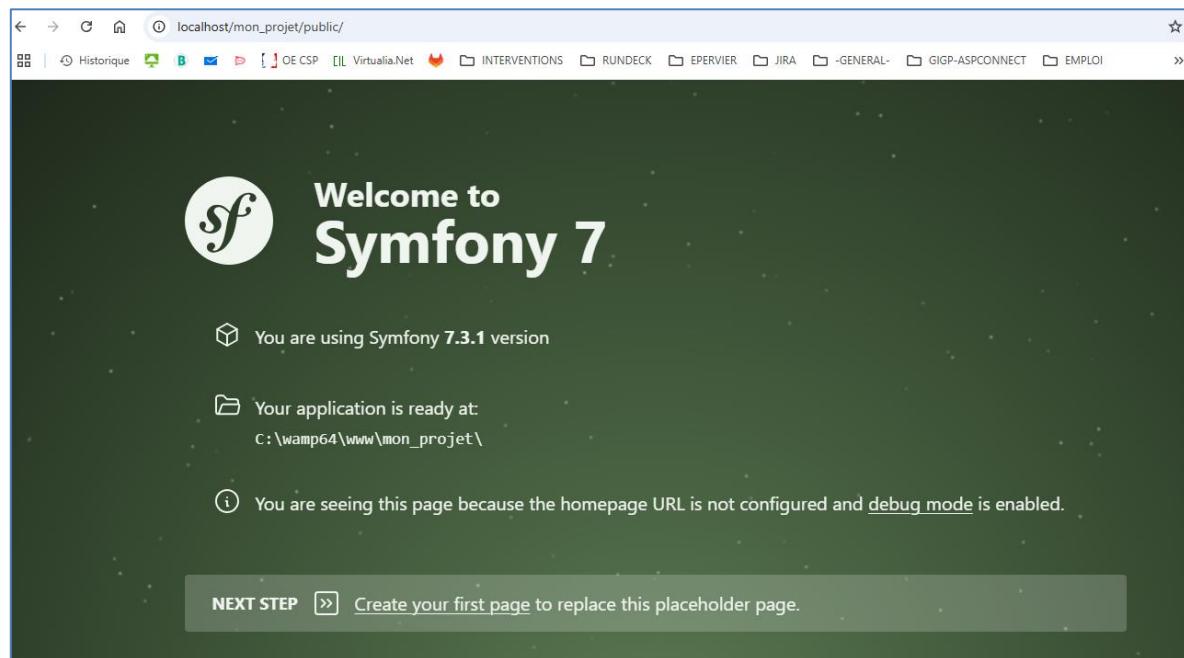
`cd mon_projet`

Nom	Modifié le	Type	Taille
mon-projet	06/07/2025 23:12	Dossier de fichiers	
wamplangues	06/07/2025 23:01	Dossier de fichiers	
wampthemes	06/07/2025 23:01	Dossier de fichiers	
favicon.ico	31/12/2010 07:40	Fichier ICO	198 Ko
add_vhost.php	02/01/2024 09:06	Fichier PHP	49 Ko
index.php	19/08/2024 15:10	Fichier PHP	30 Ko
test_sockets.php	21/09/2015 17:30	Fichier PHP	1 Ko
testmysql.php	17/06/2021 15:48	Fichier PHP	1 Ko

📖 Documentation : <https://symfony.com/doc>

Accédez directement à [http://localhost/mon\\_projet/public/](http://localhost/mon_projet/public/) dans votre navigateur.

Cette page qui s'affiche est la page d'accueil de Symfony pour toutes les applications qui viennent d'être créées.



[http://127.0.0.1/mon\\_projet/public/](http://127.0.0.1/mon_projet/public/)

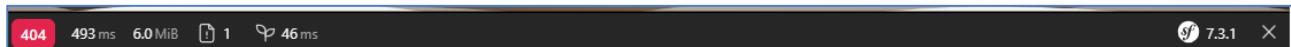


## 2.3 Activer le Profiler de Symfony

Le **Profiler** de Symfony est un outil de débogage et d'analyse des performances qui collecte des informations détaillées sur l'exécution de votre application.

C'est un système qui capture et stocke des données sur chaque requête HTTP traitée par votre application Symfony. Il fournit une interface web pour examiner ces données de manière détaillée.

Le profiler permet d'utiliser la Web Debug Toolbar, une barre d'outils qui apparaît en bas de chaque page en mode développement :



### Installation et configuration du Profiler

Le Profiler est généralement installé automatiquement dans l'environnement de développement.

Si ce n'est pas le cas, installez-le :

```
cd C:\wamp64\www\mon_projet           → se placer dans son répertoire projet  
composer require --dev symfony/profiler-pack
```

```
PS C:\wamp64\www\mon_projet> composer require --dev symfony/profiler-pack  
.composer.json has been updated  
Running composer update symfony/profiler-pack  
Loading composer repositories with package information  
Restricting packages listed in "symfony/symfony" to "7.3.*"  
Updating dependencies  
Lock file operations: 6 installs, 0 updates, 0 removals  
- Locking symfony/profiler-pack (v1.0.6)  
- Locking symfony/translation-contracts (v3.6.0)  
- Locking symfony/twig-bridge (v7.3.0)  
- Locking symfony/twig-bundle (v7.3.1)  
- Locking symfony/web-profiler-bundle (v7.3.1)  
- Locking twig/twig (v3.21.1)  
Writing lock file  
Installing dependencies from lock file (including require-dev)  
Package operations: 6 installs, 0 updates, 0 removals  
- Downloading twig/twig (v3.21.1)  
- Downloading symfony/translation-contracts (v3.6.0)
```

Assurez-vous que le Profiler est activé dans votre fichier config/packages/[dev]/web\_profiler.yaml :

yaml

```
web_profiler:  
    toolbar: true  
    intercept_redirects: false
```

Le profiler se configure dans le fichier config/packages/dev/web\_profiler.yaml. Vous pouvez

personnaliser quels collecteurs de données activer, définir des conditions d'activation, ou modifier l'apparence de la Toolbar.

Dans config/packages/dev/framework.yaml, vérifiez que le profiler est activé :

yaml

```
framework:  
    profiler:  
        only_exceptions: false
```

Assurez-vous que vous êtes bien en environnement de développement, dans le fichier .env

APP\_ENV=dev

Et/ou commande `php bin/console --version`

[Cf. partie Environnements Symfony de ce TP](#)

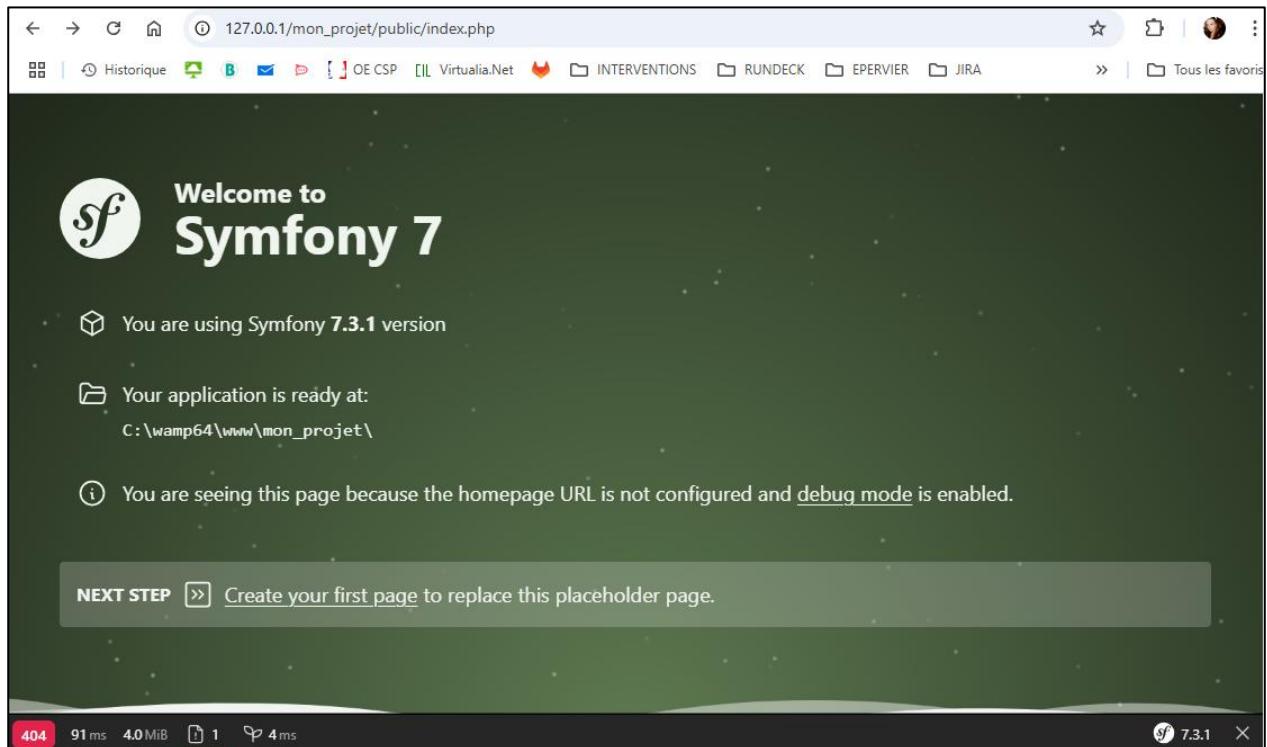
Vider le cache (si nécessaire) :

PS C:\wamp64\www\mon\_projet> `php bin/console cache:clear --env=dev`

Vérifiez que le bundle est bien enregistré dans config/bundles.php :

```
Symfony\Bundle\WebProfilerBundle\WebProfilerBundle::class => ['dev' => true, 'test' => true],
```

La toolbar devrait maintenant apparaître en bas de vos pages web lorsque vous naviguez sur votre application en mode développement.



Vous pouvez accéder au Profiler via la Toolbar, ou bien via l'URL .../\_profiler.

The screenshot shows a browser window with the URL [127.0.0.1/mon\\_projet/public/index.php/\\_profiler/021de3?panel=request&type=request](http://127.0.0.1/mon_projet/public/index.php/_profiler/021de3?panel=request&type=request). The title bar says "Symfony Profiler". The main content area displays an "ERROR 404 Not Found" message for the GET request [http://127.0.0.1/mon\\_projet/public/index.php/](http://127.0.0.1/mon_projet/public/index.php/). Below the error message, it shows the IP: 127.0.0.1, Profiled on: July 11, 2025 at 12:13:47 PM, and Token: 021de3. The left sidebar has a "Request / Response" tab selected, showing sections for Request, Response, Cookies, Session, Flashes, and Server Parameters. Under Request, there are tabs for GET Parameters, POST Parameters, and Uploaded Files, all showing "None". The Request Attributes section shows a key-value pair: \_stopwatch\_token: "a611f0". The Request Headers section shows a key-value pair: accept: "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.7".

Familiarisez-vous avec l'utilisation de la Toolbar et le Profiler.

## 2.4 Configuration

### Alias, Virtuals Hosts & Fichiers .htaccess Apache

Actuellement seule l' url [http://127.0.0.1/mon\\_projet/public/index.php](http://127.0.0.1/mon_projet/public/index.php) fonctionne (et [http://127.0.0.1/mon\\_projet/public/](http://127.0.0.1/mon_projet/public/) mais identique).

```
C:\wamp64\www\
└─ mon_projet_symfony\
    ├─ bin/
    ├─ config/
    ├─ public/           ← Point d'entrée web ACTUEL
    │   └─ index.php
    │   └─ .htaccess
    ├─ src/
    ├─ templates/
    ├─ var/
    └─ vendor/
```

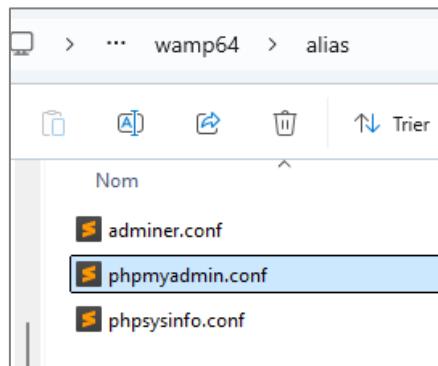
Or nous aimerais avoir la route suivante comme point d'entrée de l'application : [http://127.0.0.1/mon\\_projet/](http://127.0.0.1/mon_projet/)

Pour l'instant cette url donne :

Index of /mon_projet			
Name	Last modified	Size	Description
Parent Directory		-	
bin/	2025-07-10 11:47	-	
composer.json	2025-07-18 12:24	2.2K	
composer.lock	2025-07-18 12:24	203K	
config/	2025-07-10 11:47	-	
migrations/	2025-07-18 20:06	-	
public/	2025-07-18 19:58	-	
src/	2025-07-18 10:47	-	
symfony.lock	2025-07-18 12:12	5.1K	
templates/	2025-07-18 10:47	-	
test.php	2025-07-18 19:57	576	
var/	2025-07-10 11:48	-	
vendor/	2025-07-18 12:24	-	

### Méthode 1 - Créer un alias (recommandé si suffisant)

Mettre en place un alias devrait être suffisant. Les alias sont définis dans le répertoire éponyme de WAMP :

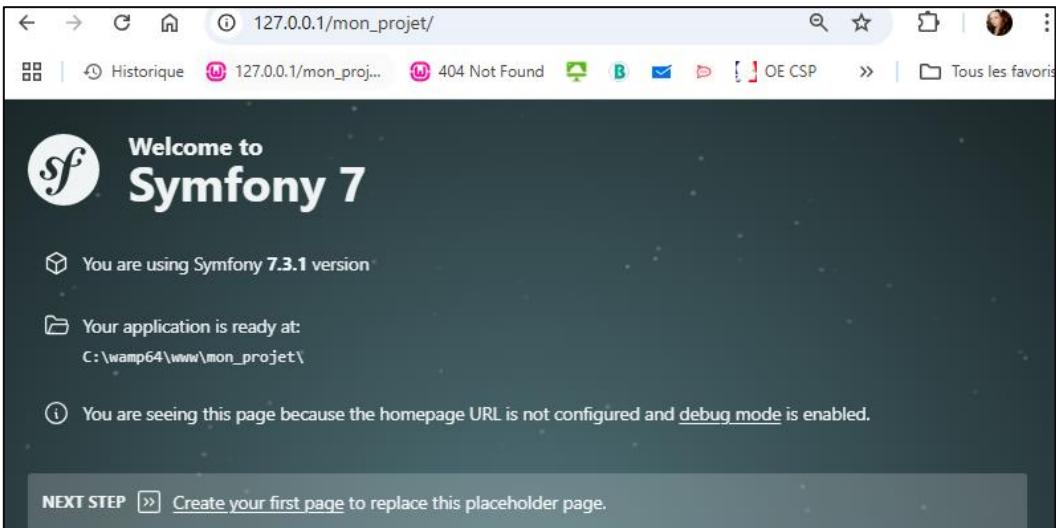


Documentation : [https://httpd.apache.org/docs/2.0/mod/mod\\_alias.html#alias](https://httpd.apache.org/docs/2.0/mod/mod_alias.html#alias)

Créer votre fichier :

```
▶ monprojet.conf •
1 Alias /mon_projet "${INSTALL_DIR}/www/mon_projet/public"
2
3 <Directory "${INSTALL_DIR}/www/mon_projet/public/">
4     Options +Indexes +FollowSymLinks
5     AllowOverride all
6     Require local
7 </Directory>
8 |
```

Puis redémarrer les services WAMP (via Clic gauche sur l'icône WAMP > Redémarrer les services) pour qu'il soit pris en compte.



[http://127.0.0.1/mon\\_projet/](http://127.0.0.1/mon_projet/)

## Méthode 2 – Définir un Virtual Host (pour information)

☞ Renseignez-vous sur les Virtual Hosts et comment en créer un.

Renseignez-vous sur les Virtual Hosts et comment en créer un.

**Virtual Hosts =>**

- Activer le module **rewrite** d'Apache si non fait :

Clic gauche sur l'icône **WAMP** (de votre barre des tâches) > **Apache** > **Modules Apache** > Cochez **rewrite\_module** > Redémarrez les services WAMP

- Créer un VirtualHost :

Clic gauche sur l'icône WAMP > Vos

```
httpd-vhosts.conf
1 # Virtual Hosts
2 #
3 <VirtualHost *:80>
4   ServerName localhost
5   ServerAlias localhost
6   DocumentRoot "${INSTALL_DIR}/www"
7   <Directory "${INSTALL_DIR}/www/">
8     Options +Indexes +Includes +FollowSymLinks +MultiViews
9     AllowOverride All
10    Require local
11  </Directory>
12 </VirtualHost>
13 #
14 <VirtualHost *:80>
15   ServerName monprojet
16   ServerAlias monprojet
17   DocumentRoot "${INSTALL_DIR}/www/mon_projet/public"
18   <Directory "${INSTALL_DIR}/www/mon_projet/public/">
19     Options +Indexes +Includes +FollowSymLinks +MultiViews
20     AllowOverride All
21     Require local
22   </Directory>
23 </VirtualHost>
24 #
```

**Virtual Host > Gestion des Virtuals Hosts ([http://localhost/add\\_vhost.php?lang=french](http://localhost/add_vhost.php?lang=french))**

Ou manuellement : éditer le fichier

<C:\wamp64\bin\apache\apache2.4.62.1\conf\extra\httpd-vhosts.conf>

Exemples :

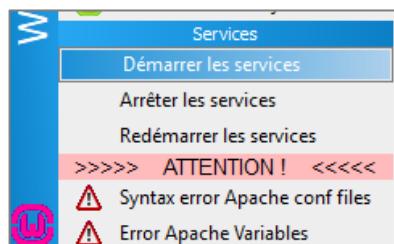
```

<VirtualHost *:80>
    ServerName mon-projet.local
    DocumentRoot "C:/wamp64/www/mon_projet/public"
    <Directory "C:/wamp64/www/mon_projet/public">
        AllowOverride All
        Require local
    </Directory>
</VirtualHost>

```

Ensuite il convient de redémarrer les services WAMP (via Clic gauche sur l'icône WAMP > Redémarrer les services) (Ou, dans notre cas, Clic droit sur l'icône WAMP > Tous > Outils > Redémarrage DNS devrait suffire).

En cas d'erreur de syntaxe, WampServer peut vous l'indiquer :



### Méthode 3 : via Fichier .htaccess

☞ Renseignez-vous sur les Fichier .htaccess, leurs intérêts et leur syntaxe.

Documentation : <https://httpd.apache.org/docs/2.4/fr/howto/htaccess.html>

### Fichier .htaccess =>

---



---



---



---

### Utilisation et intérêts =>

---



---



---



---

Créez un fichier .htaccess dans le dossier racine C:\wamp64\www\mon\_projet\ avec un contenu similaire à :

```

RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ public/$1 [L]

```

Cela devrait vous permettre d'accéder au projet via [http://127.0.0.1/mon\\_projet/](http://127.0.0.1/mon_projet/)

## Environnements Symfony

Vous n'avez qu'une seule application, mais il est probable que vous ayez besoin qu'elle se comporte différemment à différents moments. Par exemple durant la phase de développement, vous aurez besoin de tout logguer et d'avoir de bons outils de débogage. Et après le déploiement en env. de production, vous voudriez que cette même application soit optimisée pour la vitesse et enregistre uniquement les erreurs.

Symfony permet cela en définissant proposant plusieurs environnements, qui sont, par défaut :

- **dev** pour le développement local,
- **prod** pour la production (application « finale »),
- **test** for tests automatisés

Remarque : il est possible de créer soit même d'autres environnements.

Vous pouvez définir des configurations spécifiques à chacun de ces environnements.

Cependant à chaque instant, un seul environnement est utilisé, l'environnement actif, qui est défini dans le fichier .env.

```
# .env (or .env.local)
APP_ENV=dev
```

Laissez l'env. de dev. activé pour l'instant et tant qu'on sera en phase de développement (logique...)

## Activer le mode DEBUG

Méthode 1 - Via la variable d'environnement (recommandé)

```
# .env (or .env.local)
# set it to 1 to enable the debug mode

APP_DEBUG=1
```

OU Méthode 2 - Via le fichier de configuration

Yaml

```
# config/packages/framework.yaml

framework:
    debug: true
```

Le mode DEBUG de Symfony offre des informations détaillées sur les erreurs avec des stack traces complètes, facilitant grandement le diagnostic des problèmes.

 **Attention** : En production, le mode DEBUG impérativement être désactivé pour des raisons de sécurité et de performance.

## Conclusion

Les avantages de l'utilisation de WAMP sont nombreux :

- Installation simple avec interface graphique
- Gestion centralisée des services web
- phpMyAdmin intégré pour notre BDD MySQL

Cette configuration vous donne un environnement de développement Symfony complet et fonctionnel sur Windows, parfait pour commencer vos projets web.

Mais il existe des alternatives, telles que :

- **Docker** avec des conteneurs pour un environnement plus portable
- **Symfony CLI** qui inclut un serveur de développement optimisé

 Documentation : [https://symfony.com/doc/current/setup/symfony\\_cli.html](https://symfony.com/doc/current/setup/symfony_cli.html)

*Télécharger le Symfony CLI sur <https://symfony.com/download> pour installer un serveur de développement web et utiliser la commande `php bin/console server:run` pour le démarrer.*

- **XAMPP**, alternative à WampServer

 Documentation : <https://fr.wikipedia.org/wiki/XAMPP>

## [TP 1] Annexe 1 : Commandes utiles

### Installer/créer un projet Symfony

#### via Composer

```
composer create-project symfony/skeleton mon_projet
```

```
composer create-project symfony/skeleton:7.3.*-dev mon_projet
```

#### Autre possibilités (sans Composer) :

```
# run this if you are building a traditional web application  
$ symfony new my_project_directory --version="7.3.x" --webapp  
  
# run this if you are building a microservice, console application or API  
$ symfony new my_project_directory --version="7.3.x"
```

La commande « symfony new --webapp [nom du projet] » permet de créer un nouveau projet Symfony incluant tous les composants généralement nécessaires à une application Web standard, alors que la commande « symfony new [nom du projet] » ne créerait qu'un projet minimaliste.

### Voir les commandes disponibles

```
php bin/console
```

### Connaître sa version de Symfony

```
php bin/console --version
```

### Mettre à jour composer, les dépendances

```
composer update
```

### Vider le cache

```
php bin/console cache:clear --env=dev
```

### Debug config

```
php bin/console debug:config
```

### Obtenir de l'aide sur une commande

```
<commande> --help
```

Par exemple : `php bin/console debug:config --help`