

TWIG/MONOLOG

disponible aussi sur [github](#)

Les prés requis :

- PHP (minimum 7.4.24 par sécurité <https://www.php.net/downloads.php>)
- Composer (<https://getcomposer.org/download/>)

rappel :

Monolog permet de créer des logs grâce auquel on peut garder une trace de certaines actions faites par les utilisateurs ou le code.

Twig est un moteur de templates pour le langage de programmation PHP, utilisé par défaut par le framework Symfony.

Les prés requis :	1
rappel :	1
Pour commencer :	2
Installation des packages :	3
Créer les dossiers src, template et log→	3
Installation du big brother (monolog) :	4
Installation de twig :	5
Créer le fichier .html.twig→	5
On importe les classes :	6
Dans la (longue :) vue :	7
Résultat →	8

Pour commencer :

Cette commande permet de générer le fichier Composer.json.

```
composer init
```

Pour être sûr d'avoir les packages de monolog et twig, il faut les réclamer :

```
composer require monolog/monolog  
composer require twig/twig
```

Si cela ne fonctionne pas, créer manuellement le fichier Composer.json et y mettre le code suivant (en adaptant bien sur nom/prénom/sources/si besoin la licence)

```
{  
  "name": "legryan/composer-tp",  
  "type": "project",  
  "require": {  
    "monolog/monolog": "^2.3",  
    "twig/twig": "^3.3"  
  },  
  "license": "MIT",  
  "authors": [  
    {  
      "name": "yaperson",  
      "email": "yanis.legrand.1@gmail.com"  
    }  
  ],  
  "autoload": {  
    "psr-4": {"App\\": "src/" }  
  }  
}
```

Installation des packages :

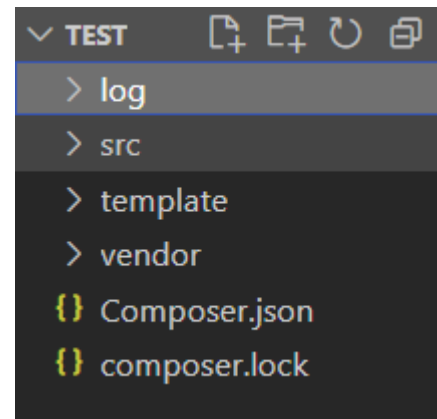
```
composer install
```

Créer les dossiers src, template et log→

src => contiendra le code php

template => contiendra les vues en .html.twig

log => contiendra les log



Installation du big brother (monolog) :

Dans un fichier php, rangé dans /src.

Ajouter le code ci dessous :

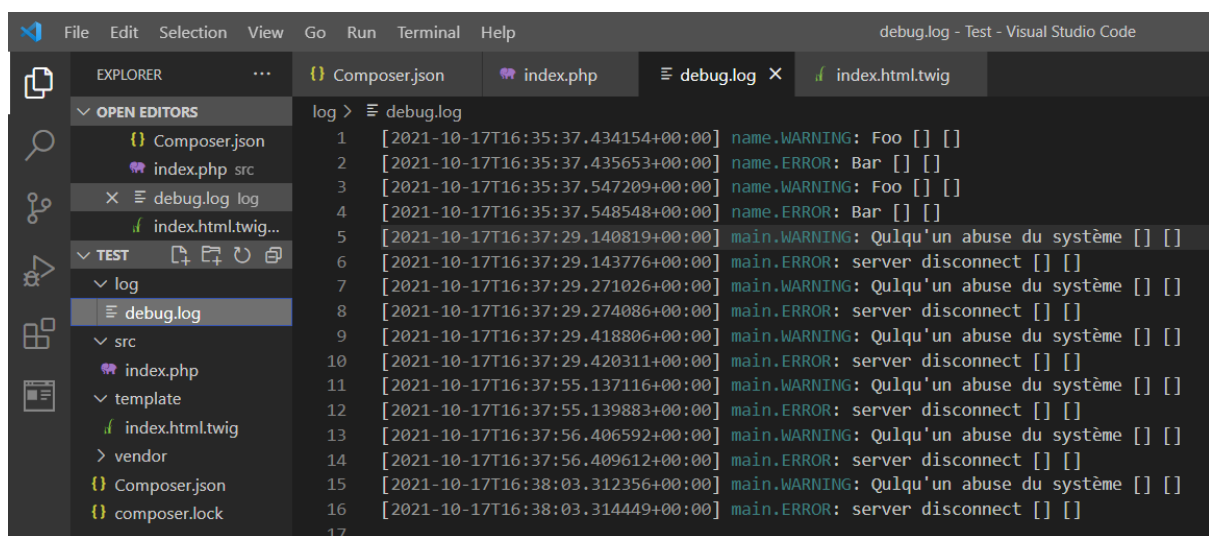
```
<?php
require_once '../vendor/autoload.php'; // fait appel à l'autoload (l'équivalent d'un
include)

use Monolog\Logger; // indique l'utilisation de la Classe Monolog\Logger
use Monolog\Handler\StreamHandler; // et la Classe Monolog\Logger\StreamHandler

// create a log channel
$log = new Logger('main');
// créer un dossier /log et un fichier debug.log
$log->pushHandler(new StreamHandler('../log/debug.log', Logger::WARNING));

// add records to the log
$log->warning('Qulqu\'un abuse du système');
$log->error('server disconnect');
```

Le résultat, à chaque fois qu'un utilisateur se connecte sur la page, un fichier debug.log est créer puis des log sont écrits :

The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left, the Explorer view in the center, and the Output view at the bottom. The Explorer sidebar shows the project structure with folders like 'log', 'src', 'template', and 'vendor'. The Explorer view shows the 'debug.log' file selected. The Output view at the bottom displays the log output, which is a list of log entries with timestamps and messages. The log entries are as follows:

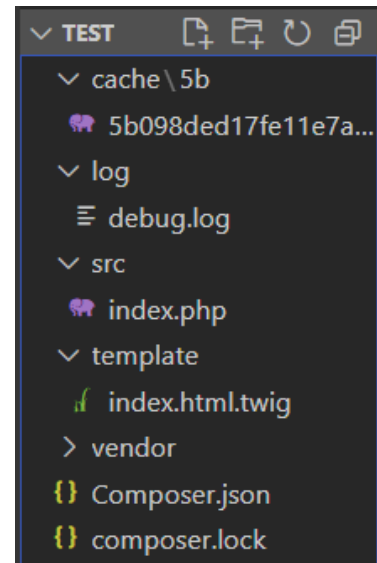
Line	Timestamp	Level	Message
1	[2021-10-17T16:35:37.434154+00:00]	name.WARNING	Foo [] []
2	[2021-10-17T16:35:37.435653+00:00]	name.ERROR	Bar [] []
3	[2021-10-17T16:35:37.547209+00:00]	name.WARNING	Foo [] []
4	[2021-10-17T16:35:37.548548+00:00]	name.ERROR	Bar [] []
5	[2021-10-17T16:37:29.140819+00:00]	main.WARNING	Qulqu'un abuse du système [] []
6	[2021-10-17T16:37:29.143776+00:00]	main.ERROR	server disconnect [] []
7	[2021-10-17T16:37:29.271026+00:00]	main.WARNING	Qulqu'un abuse du système [] []
8	[2021-10-17T16:37:29.274086+00:00]	main.ERROR	server disconnect [] []
9	[2021-10-17T16:37:29.418806+00:00]	main.WARNING	Qulqu'un abuse du système [] []
10	[2021-10-17T16:37:29.420311+00:00]	main.ERROR	server disconnect [] []
11	[2021-10-17T16:37:55.137116+00:00]	main.WARNING	Qulqu'un abuse du système [] []
12	[2021-10-17T16:37:55.139883+00:00]	main.ERROR	server disconnect [] []
13	[2021-10-17T16:37:56.406592+00:00]	main.WARNING	Qulqu'un abuse du système [] []
14	[2021-10-17T16:37:56.409612+00:00]	main.ERROR	server disconnect [] []
15	[2021-10-17T16:38:03.312356+00:00]	main.WARNING	Qulqu'un abuse du système [] []
16	[2021-10-17T16:38:03.314449+00:00]	main.ERROR	server disconnect [] []

Bravo, vous avez maintenant les base de monolog, pour plus de renseignement, rendez vous sur la doc officielle → [ici](#)

Installation de twig :

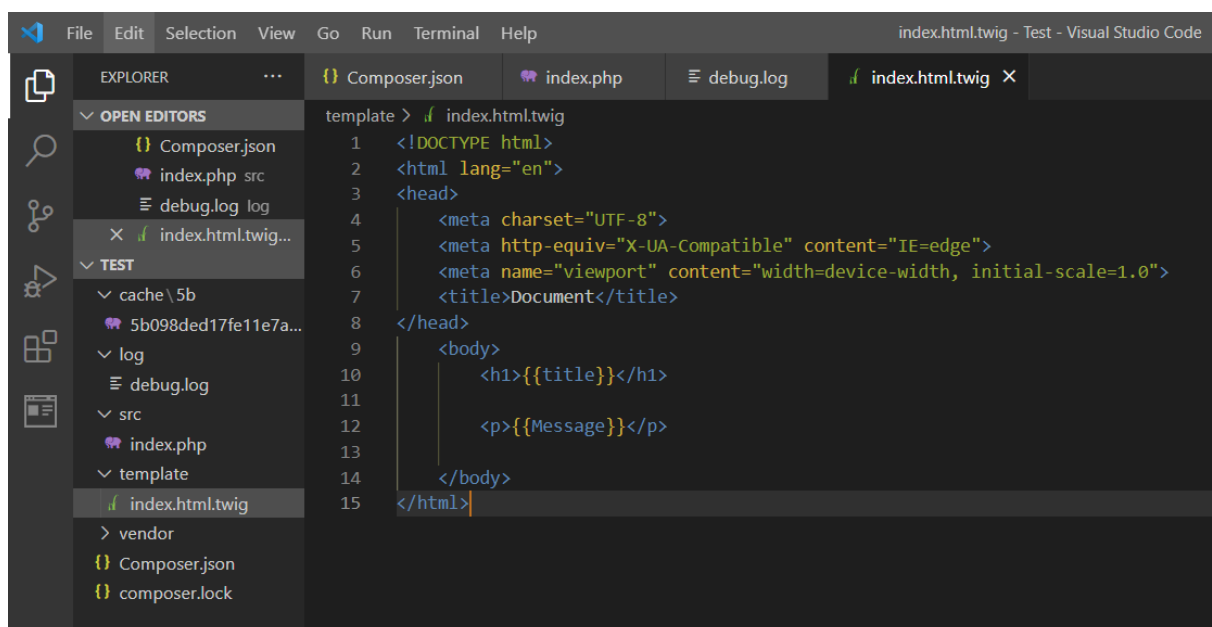
Créer le fichier .html.twig→

dans template => contiendra les vues en .html.twig



Le fichier .html.twig est un html qui sera pris en charge par twig.

La syntaxe reste celle de HTML.



On importe les classes :

Dans le même fichier php que pour monolog, ajouter les import de classes de twig :

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

use Monolog\Logger;
use Monolog\Handler\StreamHandler;
use Twig\Loader\FilesystemLoader; // on ajoute l'import des classes pour twig
use Twig\Environment;
```

```
// on créer les objets pour utiliser twig

// indique l'emplacement des vues
$loader = new FilesystemLoader('../template');

// créer un dossier cache
$twig = new Environment($loader, ['cache' => '../cache']);

try {
    $message = "Mon message !";
    echo $twig->render('index.html.twig', [ //indique quel vue on utilise
        'title' => 'Liste des utilisateurs', // un titre
        'Message' => $message, // un message
        // on peut ajouter ce que l'on veut
        // 'desChiffres' => '0 1 2 3 4 5...',
    ])
};
} catch(PDOException $e) {
    print('erreur de connection : ' . $e->getMessage());
}
```

Dans la (longue :) vue :

en respectant la syntaxe `<balise> {{ desChiffres }} </balise>` on peut afficher dynamiquement le contenu du php dans des balise html, sans mélanger les deux grâce à twig

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

    <body>

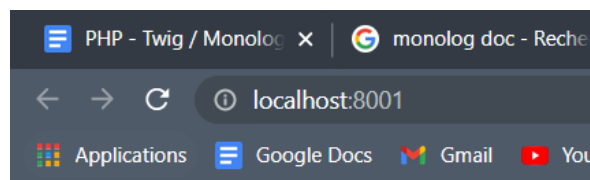
        <h1>{{title}}</h1>

        <p>{{Message}}</p>

    </body>

</html>
```

Résultat →



Liste des utilisateurs

Mon message !