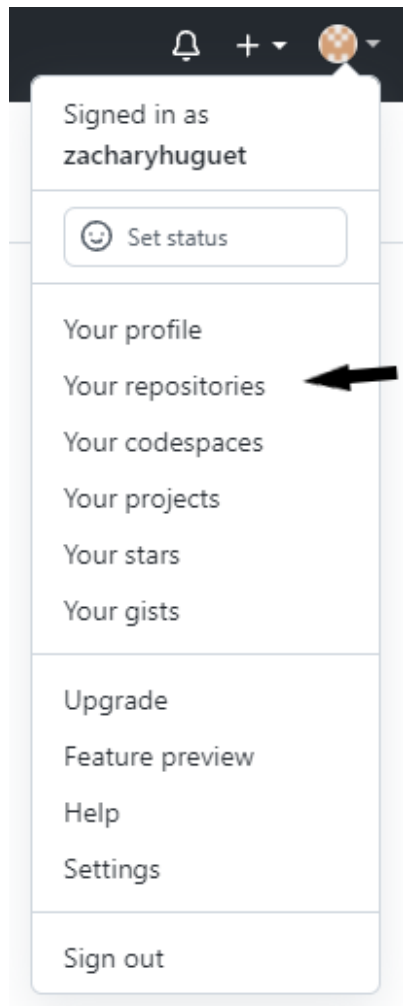


Utilisation Composer et Twig

Création d'un dépôts GitHub	2
Installation php	5
Installation Symfony	6
Installation Composer Monolog et Twig	7
Twig	9
Création d'une base de donnée	10
PHP Classe et gestion d'utilisateur	11
Utilisation d'un bootstrap	13

Création d'un dépôts GitHub

On va sur <https://github.com/> ensuite on va sur son profil et on clique sur "Your repositories" pour créer un dépôt.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *



zacharyhuguet ▾

Repository name *

/

Great repository names are short and memorable. Need inspiration? How about **bug-free-octo-umbrella**?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.



Add a README file

This is where you can write a long description for your project. [Learn more.](#)



Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)



Choose a license

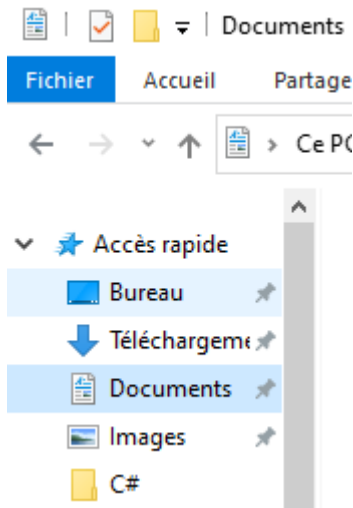
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

Pour la création on indique le nom du dépôt et on le met en publique pour qu'il soit visible par tous le monde.

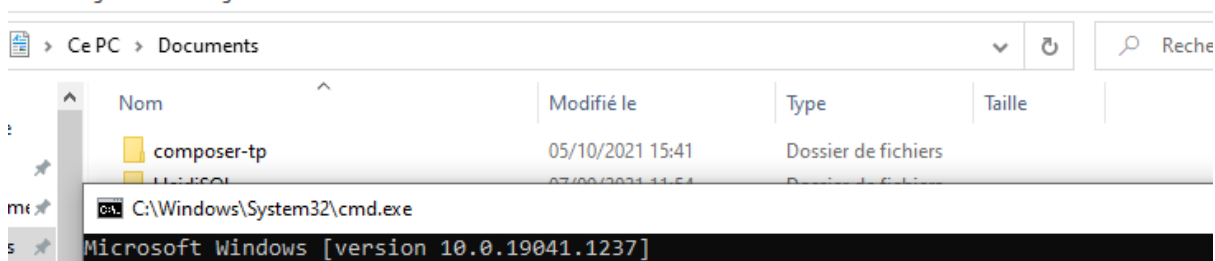
ensuite on accède au dépôts en cliquant dessus puis on va cliquer sur "Code" (le bouton vert) pour copier le lien et cloner le dépôts

The screenshot shows the GitHub interface for a repository named 'HUGUET Zachary indexFais'. The top navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the navigation bar, the repository name is displayed along with '1 branch' and '0 tags'. A table lists the files in the repository: 'cache/51' and 'src', both pointing to 'indexFais'. On the right side, there is a 'Code' button (green) and a 'Clone' button (grey). A dropdown menu is open under the 'Code' button, showing the 'HTTPS' option selected. The URL 'https://github.com/zacharyhuguet/composer' is displayed, and a black arrow points to it. Below the URL, it says 'Use Git or checkout with SVN using the web URL.'



On va ensuite dans ces documents

Puis dans la barre pour accéder le chemin on tape "CMD"



Dans ce CMD on tape la commande "git clone (+ le lien que vous venez de cloner)".

```
C:\Windows\System32\cmd.exe
Microsoft Windows [version 10.0.19041.1237]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\huguzac\Documents>git clone https://github.com/zacharyhuguet/composer-tp.git
```

Le pc vient de cloner le dépôt github vide que vous avez créé précédemment.
Comme ça le dossier du pc et le dépôts github seront lié et vous pourrez mettre à jour vos document facilement avec les commandes :

- git add .
- git commit -m "(nom de la mise à jour)".
- git push

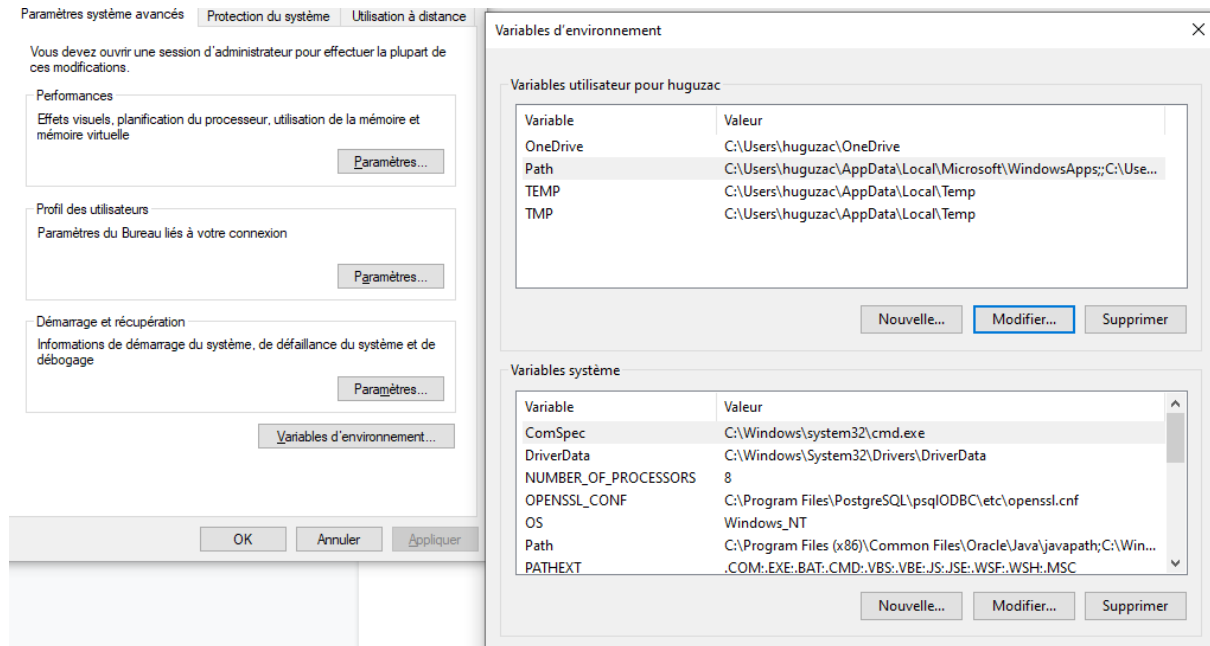
On doit aussi créer un fichier .gitignore pour indiquer à git lors d'un push quel fichier il ne doit pas prendre (fichier avec les mots de passe de connexion à la BD on crée donc un conf.sample pour indiquer comment on fait la connexion à la BD).

Installation php

Dernière version stable 8.0.11.

Pour télécharger php on va sur <https://www.php.net/downloads>

on indique le chemin du php.ini dans les variable d'environnement puis dans le path



Modifier la variable d'environnement



%USERPROFILE%\AppData\Local\Microsoft\WindowsApps

C:\Users\huguzac\AppData\Local\Programs\Microsoft VS Code\bin

Nouveau

Modifier

Parcourir...

Supprimer

Déplacer vers le haut

Déplacer vers le bas

Modifier le texte...

OK

Annuler

Installation Symphony

<https://symfony.com/download>

On télécharge et exécute Symphony-Setup.exe

php C:\composer\v2\composer.phar create-project symfony/website-skeleton:"^4.4" essai1

On indique le chemin jusqu'à la localisation du composer.phar

On indique la version que l'on veut et le nom ensuite.

Installation Composer Monolog et Twig

On doit d'abord télécharger Composer <https://getcomposer.org/download/>
Moi j'ai installé la version 1.10.10

Ensuite on lance un document avec un logiciel de programmation (Visual Studio Code) et on on un nouveau terminal dans celui-ci.

Dans celui ci on commence par taper la commande :

- composer init

Ensuite on indique les champ suivant :

- Package name : VOTRE-NOM/VOTRE-NOM-DE-DÉPÔT
- Description : Small project to learn the composer tool
- Author : VOTRE-NOM <VOTRE-ADRESSE-EMAIL>
- Minimum Stability : stable
- Package Type : project
- License : MIT
- Would you like to define your dependencies : no
- Would you like to define your dev dependencies : no
- Do you confirm generation : yes

Après ses champs renseigner on tape la commande :

- composer install

Ainsi Composer va nous générer des documents comme composer.json

Ensuite on installe Monolog avec la commande :

- composer require monolog/monolog

Monolog sert à tracer le bon fonctionnement d'un code (savoir si tout marche bien et si non renvoi un erreur) il y à plusieurs :

- DEBUG
- INFO
- NOTICE
- WARNING
- ERROR
- CRITICAL
- ALERT
- EMERGENCY

Et on installe aussi Twig avec la commande:

- `composer require "twig/twig"`

A la fin le `composer.json` doit ressembler a quelque chose comme ça.

```
composer.json X
} composer.json > {} require > twig/twig
1  {
2      "name": "huguzac/composer-tp",
3      "description": "Small project to learn t
4      "type": "project",
5      "license": "MIT",
6      "authors": [
7          {
8              "name": "HUGUET Zachary",
9              "email": "huguzac@ndlp.fr"
10         }
11     ],
12     "minimum-stability": "stable",
13     "require": {
14         "monolog/monolog": "^2.3",
15         "twig/twig": "^3.3"
16     },
17     "autoload": {
18         "psr-4": {"App\\": "src/"}
19     }
20 }
21
```

Après, on créer à la racine un dossier `src` pour contenir les fichiers sources de notre projet PHP.

et on crée aussi un dossier `templates` pour contenir les fichiers de nos modèles de pages HTML.

```
> src
> templates
> vendor
◆ .gitignore
{} composer.json
{} composer.lock
```

Ensuite on doit ajouter la ligne `autoload` dans le `composer.json`:

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


Pour indiquer où il ira chercher les fichiers PHP

Après, dans le dossier templates on crée le fichier html.twig

dans l'index.php:

```
echo $twig->render(
    'user/index.html.twig',
    [
        'users' => $users,
        'title' => "Titre de ma page très géniale",
    ]
);
```

et dans le html.twig:

```
<body>
    <h1>Logiciel - {{ title }}</h1>

    <ul>
        {% for user in users %}
            <li>{{ user.email }}</li>
        {% endfor %}
    </ul>
```

On voit que l'on initialise des variables dans le PHP et que l'on peut les appeler dans html.twig sans utiliser les balises pour ouvrir et fermer le php tout le temps.

Pour lancer le localhost on fait la commande:

- php -S localhost:8000 -t src

on ajoute le "-t src" pour dire que les fichiers de base à afficher sont dans le dossier src.

Twig

Les Fonctionnalités de twig:

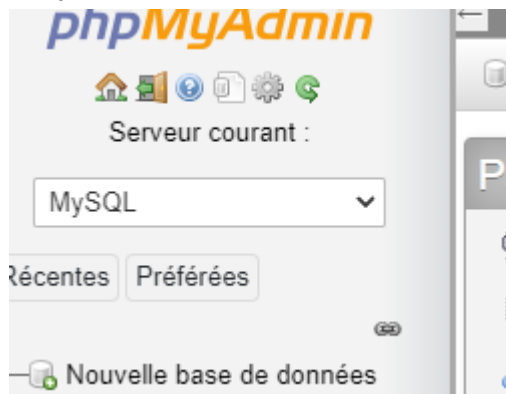
- {{ variable }} : affiche quelque chose
- {% commande %} : fait quelque chose
- {# commentaire #} : n'affiche rien et ne fait rien

Exemples de commandes :

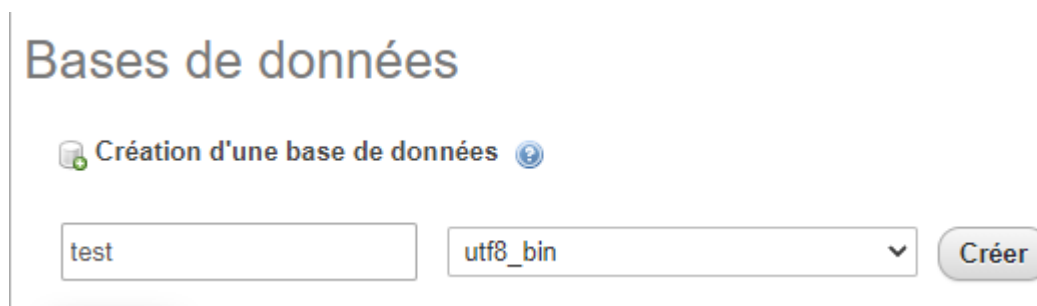
- Pour l'affichage: <h1>Logiciel - {{ title }}</h1>
- {% for user in users %} -> boucle
- {# Un commentaire #}

Création d'une base de donnée

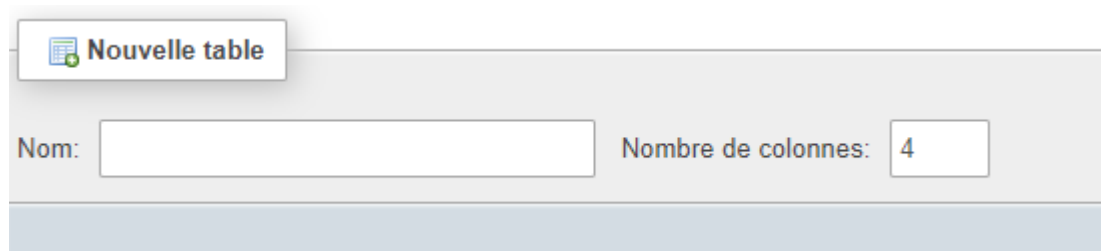
Moi j'ai créé ma base de donnée avec phpmyadmin de wamp



On clique donc sur “Nouvelle base de données”.



Ensuite on lui indique son nom et son type de langage (nous utf8)



Après on lui indique le nom de la table et le nombre de colonne que l'on veut dedans
La pour la gestion d'utilisateur je vais en mettre 4:

- id (clée primaire)
- email (varchar)
- password (varchar) -> sha1 pour crypter le mot de passe
- groupe (varchar)

On peut maintenant créer un utilisateur.
et renseigner au php comment se connecter à la base de données.

pour ça on créer un conf.php dans lequel on rentre:

```
confsample.php
<?php
$dsn = 'mysql:dbname=NomDeLaBase;host=Ip';
$user = 'UnUtilisateur';
$password = 'UnMotDePasseTrèsSecurisé';
```

PHP Classe et gestion d'utilisateur

```
wamp64 > www > php-poo > archer.php
k?php

final class Archer extends Personnage
{
    public function attaquer(Personnage $persoAFrapper): Personnage
    {
        $persoAFrapper->_degats += $this->_force;
        return $this;
    }

    public function insulter()
    {
        print("</br> tu vise comme un pied! ");
    }
}
```

Ici on crée une classe Archer avec un final qui veut dire qu'elle n'aura pas d'héritage on lui donne 2 méthodes attaquer et insulter

On crée ensuite un fichier PersonnageManager pour gérer les personnages

```
class PersonnagesManager
{
    private $_db;

    public function __construct(PDO $db)
    {
        $this->setDb($db);
    }

    public function setDb(PDO $db): PersonnagesManager
    {
        $this->_db = $db;
        return $this;
    }
}
```

Dedans on met le Constructeur qui aidera à la création de nouveaux personnages.

On crée après un nouveau fichier avec une classe Personnage ou l'on renseigne tous les attributs d'un personnage et les assesseurs et les guetteurs pour chacun des attributs.

```
abstract class Personnage
{
    // Déclaration des attributs et méthodes ici.
    private $_id = 0;
    private $_nom = 'Inconnu'; // Son nom, par défaut 'Inconnu'.
    protected $_force = 50; // La force du personnage, par défaut à 50.
    private $_experience = 1; // Son expérience, par défaut à 1.
    private $_degats = 0; // Ses dégats par défaut
    private $_niveau = 0;
    private $_classe = 0;
    private $_poche = 50;
}
```

```
public function setNom(string $nom): Personnage
{
    if (!is_string($nom)) // S'il ne s'agit pas d'un texte.
    {
        trigger_error('Le nom d\'un personnage doit être un texte', E_USER_ERROR);
        return $this;
    }
    $this->_nom = $nom;
    return $this;
}

public function getNom(): string
{
    return $this->_nom;
}
```

Utilisation d'un bootstrap

On cherche un thème souhaité sur <https://getbootstrap.com/>

Une fois trouvé on copie et on le met dans de le <head>

```
<!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">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css" rel="stylesheet" in
  <script src="https://kit.fontawesome.com/2696450378.js" crossorigin="anonymous"></script>
```