

FIL CONDUCTEUR DE SUPERBLOG

Introduction

« Superblog » est le projet sur lequel nous allons travailler régulièrement tout au long de la seconde partie de l'année. Pour commencer en PHP, par la suite, nous intégrerons du JavaScript

Comme son nom l'indique, enfin, je l'espère :), « Superblog » est un moteur allégé de type gestionnaire de contenu ou CMS destiné à la création de Blog (comme son très grand frère WordPress).

Je vais vous guider dans la réalisation de ce projet. Mais allez devoir mettre les mains dans le cambouis, sinon ce ne serait pas marrant !

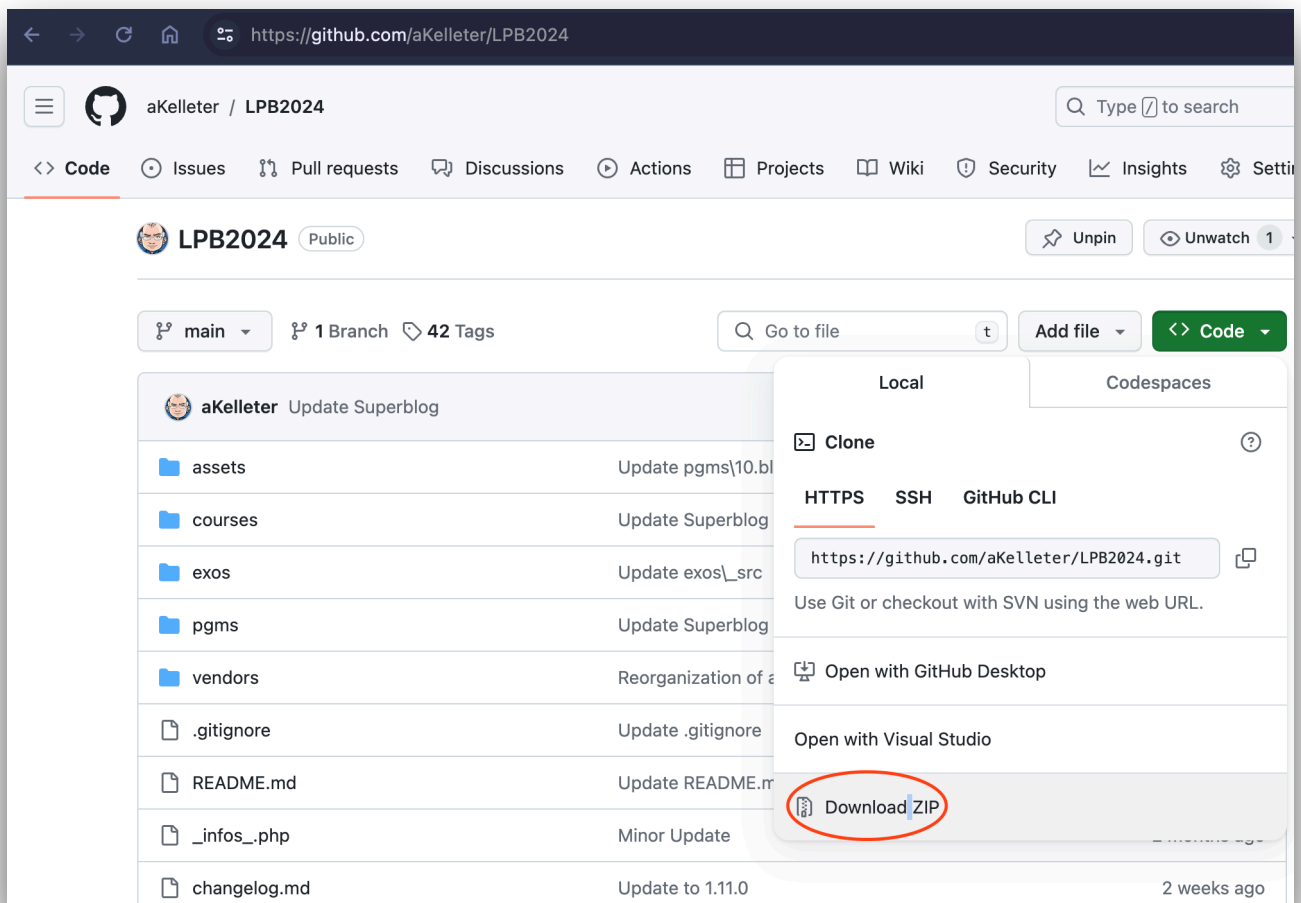
Nous avons déjà abordé et commencé à écrire les premières pages et scripts du projet « Superblog » mais pour la plupart d'entre vous cela s'est un peu passé dans la douleur.

C'est la raison d'exister de ce document !

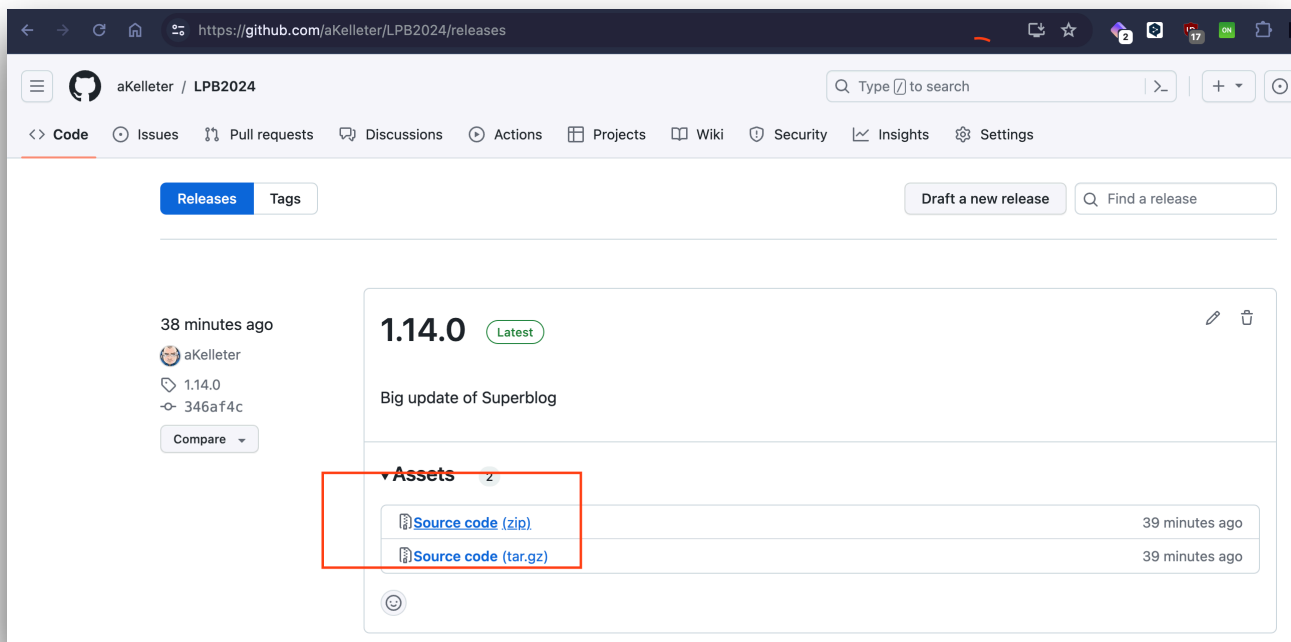
1. Première chose, le code de Superblog se trouve dans l'application LPB2024, plus exactement ici : LPB2024 / pgms / 10.superblog /

Comme vous devez le savoir maintenant, le code source de LPB2024 est disponible sur GitHub. Autrement dit, pour accéder au code source, qui est en permanence mis à jour en fonction de nos avancées dans le projet, vous devez soit :

- Utiliser GitHub pour récupérer la dernière mise à jour.
- Aller sur GitHub et récupérer le code source au format ZIP (voir image ci-dessous - Cliquez sur le bouton « <> Code » (en vert))



- Aller sur GitHub et récupérer la dernière Release de LPB2024

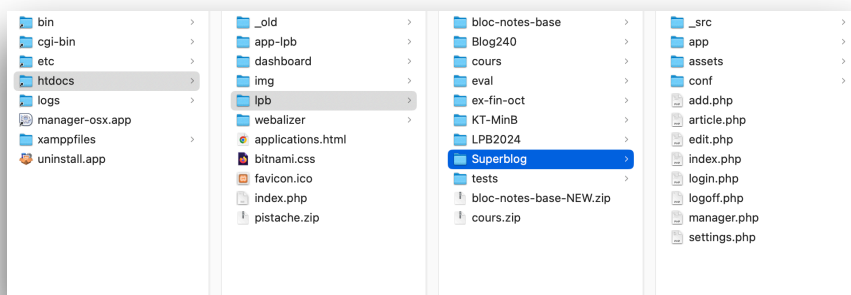


2. Seconde chose, vous devez faire une copie du répertoire de Superblog qui se trouve dans LPB2024 qui se nomme **10.superblog**

Et placer cette copie dans le répertoire « htdocs » de XAMPP (voir dans htdocs/lpb) si vous avez réalisé ce type de structure sur votre ordinateur.

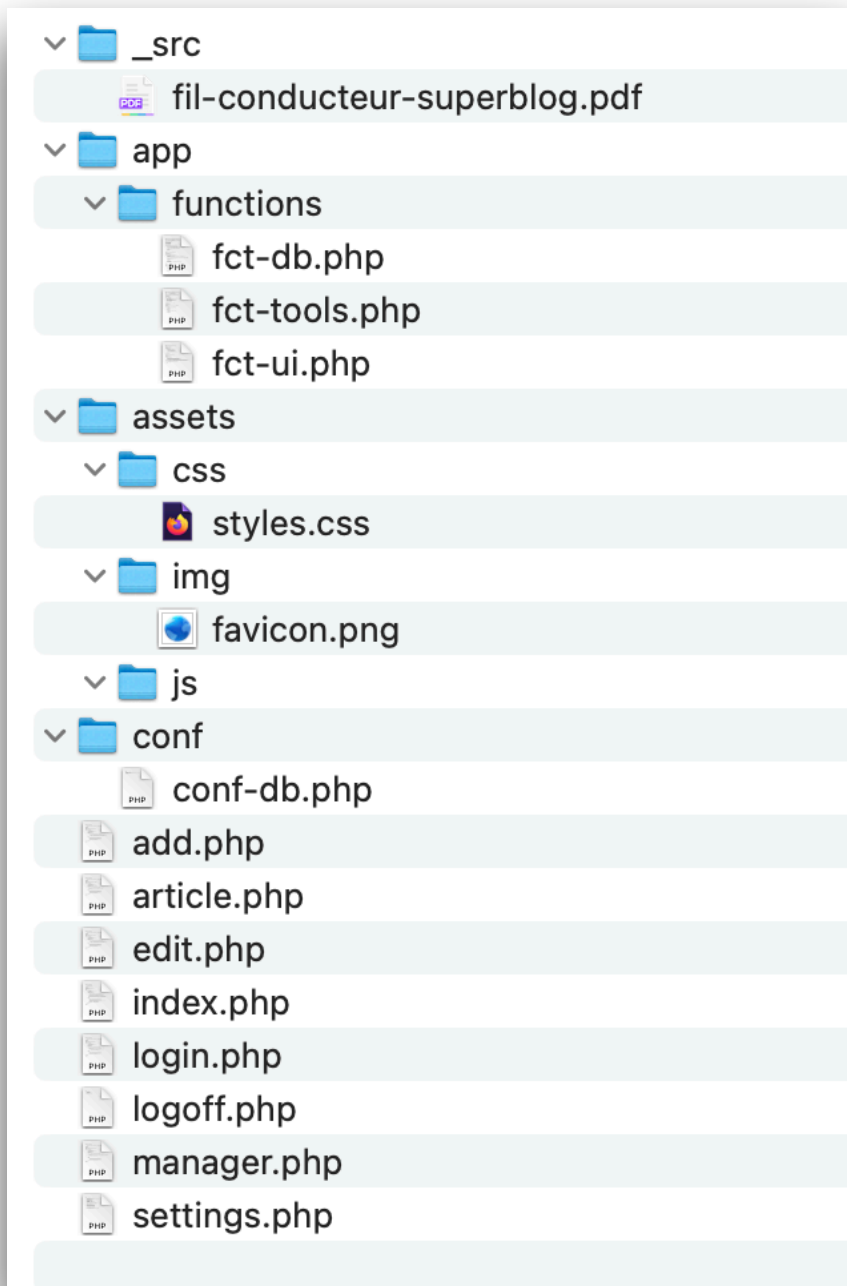
Si vous êtes déjà bien avancé dans votre projet Superblog, vous n'êtes pas obligé d'utiliser le code source du SuperBlog de LPB2024 et continuer sur votre projet. Vous pourrez cependant, si besoin est, vous inspirer de certains éléments et/ou fichiers du projet.

Si vous n'êtes pas très avancé ou si cela ressemble plus à chemin de croix qu'à une ballade dans la campagne ensoleillée, alors je vous conseille de reprendre le code source du Superblog de LPB2024.



3. La structure et le contenu

Vous trouverez ci-dessous la structure de base du projet. Elle sera à même d'évoluer dans le temps bien entendu.



Structure à la date du 13/12/2023

Le répertoire `_src/`

Il contient des fichiers complémentaires au projet, mais ils ne font pas partie de l'application « Superblog ».

Le répertoire app/

Il contient les fichiers de bases de l'application. Par exemple, les fichiers de fonctions (dans le sous-répertoire « fonctions ») et prochainement les fichiers de classes dans le sous-répertoire « classes ».

Le répertoire assets/

Il contient ce que nous nommons les « assets » ou « actifs ». En termes de conception et de développement de sites web, les « assets » font généralement référence au contenu textuel, aux règles CSS, aux graphiques, aux photographies, aux vidéos, aux fichiers audio qui fait partie de l'application, ce ne sont pas des données générées par l'application.

Le répertoire conf/

Il contient le fichier de configuration que l'utilisateur de Superblog doit mettre à jour pour faire fonctionner son application. Actuellement, il contient les données qui permettent à Superblog de se connecter à une base de données.

La racine du projet /

Elle contient tous les fichiers de base et plus exactement les pages et certains scripts de l'application.

index.php : C'est le fichier / page qui est recherché par le serveur web quand rien n'est stipulé dans l'URL, ex. : www.240am.be. C'est la page d'accueil de notre Blog, c'est elle qui va afficher la liste de nos articles **publiés** aux visiteurs.

add.php : C'est le fichier / page qui va nous permettre d'ajouter un article à notre Blog. Pour ce faire, nous allons afficher un formulaire et nous traiterons les données envoyées par ce dernier.

article.php : C'est le fichier / page que nous allons utiliser pour afficher un article en entier, c.-à-d. Son titre, son contenu et en principe d'autres éléments comme une ou plusieurs images. Cette page recevra dans son URL l'ID de l'article à afficher.

edit.php : C'est le fichier / page qui va nous permettre de mettre à jour, de modifier un article. Elle sera composée d'un formulaire semblable à celui de la page add.php et doit recevoir dans son URL l'ID de l'article à modifier.

login.php : C'est le fichier / page d'identification de l'application. Il est composé d'un mini formulaire avec un login et mot de passe.

logoff.php : C'est le script de déconnexion de l'application.

manager.php : C'est le fichier / page de gestion des articles. Comme sur la page index.php, cette page va afficher la liste des articles, y compris **ceux qui ne sont pas encore publiés**. Pour chaque article, on doit afficher des liens qui vont permettre d'éditer, d'afficher et de supprimer l'article en question.

settings.php : C'est le script de configuration de l'application proprement dit. Il contient les constantes et les éléments logiques au bon fonctionnement de l'application de façon générale.

4. La structure des fichiers

```
1  <?php
2      require_once('settings.php');
3
4      /**
5       * ICI VOUS ECRIVEZ LE CODE PHP QUI GERE LA LOGIQUE ET LES DONNEES DE L'APPLICATION
6       */
7
8  ?>
9  <!DOCTYPE html>
10 <html lang="en">
11 <head>
12     <?php displayHeadSection('Ajouter un article'); ?>
13 </head>
14 <body>
15     <div class="container">
16         <div id="header-logo">
17             <h1><?php echo APP_NAME; ?></h1>
18         </div>
19         <div id="main-menu">
20             <?php displayNavigation(); ?>
21         </div>
22         <h1>Ajouter un article</h1>
23         <div id="content">
24             <!--
25                 Créez ici un formulaire HTML pour ajouter un nouvel article
26                 * Astuces :
27                 - L'attribut "action" de votre balise form devra contenir "manager.php"
28                 - C'est dans le fichier manager.php que l'on va traiter les données du formulaire
29                 - L'attribut "method" devra contenir "post"
30             -->
31
32         </div>
33         <footer>
34             <!--
35                 Ouvrez une balise php pour lancer la fonction d'affichage
36                 du footer. Fonction que vous allez écrire dans fct-ui.php
37                 Affichez le nom de l'app sa version sa date de mise à jour
38                 et d'autres choses si vous le souhaitez
39             -->
40         </footer>
41     </div>
42 </div>
43 </body>
44 </html>
```

Les fichiers (page) .php sont avant tout des fichiers qui possèdent une structure HTML complète **<html></html>**.

Au-dessus de la structure HTML sont placées les balises **<?php ?>** indiquant qu'à cet endroit, nous allons écrire du code PHP.

Dans cet espace au-dessus des pages réservé au code PHP, nous allons écrire ce que nous appellerons le **contrôleur** de la page, autrement dit le code qui va gérer la logique, les données et autres traitements nécessaires au bon fonctionnement de la page et de l'application en général.

Dans la structure HTML, nous essayons dans la mesure du possible de ne faire que des affichages en appelant des fonctions qui retournent du code HTML tel que **<?php displayNavigation(); ?>**.

Nous retrouverons dans la structure HTML de courtes sections de code PHP tels que celle ci-dessus et rien de plus. Tout le code métier, logique et autres processus business doit se trouver dans le contrôleur en haut de page.

Créer vos fonctions d'affichage dans le fichier **fct-ui.php**. Adopter une convention de nommage similaire ou identique à celle que j'ai utilisée, c.-à-d. Par exemples :

```
function displayArticlesPublies() {}  
function displayArticlesAll() {}
```

Et ainsi de suite.

5. Des cadeaux pour Noël !

Les fichiers de fonctions, **fct-db.php** et **fct-tools.php** possèdent déjà leurs fonctions.












En ce qui concerne le fichier **fct-db.php**, toutes les fonctions sont créées et opérationnelles, vous n'avez pas à les écrire. Si vous le souhaitez, ceux qui se sont déjà penchés dessus, continuez à écrire les vôtres. Nous reviendrons dessus plus tard sur ces fonctions. Vous devez simplement les utiliser en les appelant.

Le fichier **logoff.php** est également complété.

Le fichier **styles.css** possède déjà quelques règles CSS.

Je n'ai pas intégré le framework CSS **Bootstrap** au projet, je vous laisse le choix de l'utiliser ou non, ce n'est pas nécessaire.

6. La DB

Objets		articles@blog (localhost)		users@blog (localhost)	
  		   		   	
	id	email	passwd		
	1	john@mail.com	1234		
	2	alain@kelleter.be	AZERTY		

Les champs et les données de la table **users**

Objets

articles@blog (localhost)

users@blog (localhost)

</

Les champs e les données de la table **articles**

Il est possible que vous n'ayez pas exactement le même contenu dans la table article.

Si vous avez un problème avec la base de données, j'ai placé dans le répertoire `_src` du projet un fichier **blog.sql**, vous pouvez l'utiliser pour créer ou recréer la DB blog sur le moteur DB MySQL de votre XAMPP.

Pour ce faire, rendez-vous dans la webapp [phpMyAdmin](#) et cliquez en haut de la page sur **Importer**, choisissez ensuite le fichier situé dans le répertoire `_src` du projet : `blog.sql`. Pour terminer, descendez dans le bas de la page et cliquez que le bouton **Importer**.

7. Et maintenant, que dois-je faire ?

Bien, quelle question..., toute l'application pardi !

Mais plus sérieusement, pour le lundi **18/12/2023** :

Pour celles et ceux qui ne l'ont pas encore fait...

Je vous demande de venir au cours avec une page **index.php** fonctionnelle, autrement dit une page qui affiche proprement (pas un tableau avec `var_dump()`), les articles de la base de données Blog qui a été créé sur votre ordinateur.

Nous avons terminé un peu à l'arrache l'affichage des données brutes sur la page **index.php** mais la fonction va chercher les données est déjà appelée sur la page **index.php**, il vous faut écrire la fonction d'affichage qui va lister les titres des articles sur la page. La sortie d'une DB c'est un tableau donc...

.

Ne vous arrêtez pas en si bon chemin, si vous le pouvez, commencez la page **manager.php**, qui au niveau affichage est similaire à la page **index.php**, avec des éléments en plus.

Et si vous avez du temps, faites déjà la structure HTML de la page **login.php**, Nous avons déjà fait un formulaire d'identification.

Encore du temps :) Commencez les pages **add.php** et **edit.php**, au niveau HTML ce sont des formulaires

Pour les autres...

Celles et ceux qui ont déjà fait la page **index.php**, voir d'autres pages, continuez dans cette direction et avancez dans le projet.

Analyse générale du projet

Mise à jour du : 17/12/2023

Quelles sont les pages du projet qui affichent du contenu et qui sont accessibles ?

add.php
article.php
edit.php
index.php
login.php
logoff.php
manager.php

Quelles sont les pages accessibles sans être identifié ?

article.php
index.php
login.php

Quelles sont les pages accessibles après identification ?

add.php
edit.php
manager.php

Le script de déconnexion est :

Ce n'est pas une page qui affiche du contenu, c'est un script qui exécute une série d'instruction pour déconnecter l'utilisateur identifié de l'application.

logoff.php

Le script de configuration de l'application est :

settings.php

Il contient :

- les constantes de configuration de l'application
- le lancement des sessions
- le chargement des fichiers :
 - les credentials de connexion à la DB
 - les fichiers de fonctions
- l'instanciation de la connexion à la DB -> création de l'objet connexion \$conn

Remarque : ce fichier sera inclus dans toutes les pages du projet (voir page du projet accessible (en bleu))

Analyse du fichier : index.php

C'est un fichier de type page qui affiche du contenu à l'écran

C'est la page d'accueil du site

C'est la page index du site

C'est une page publique

La page possède une structure HTML complète

La structure HTML est basique, cela signifie que les éléments suivants seront affichés à l'aide de fonctions PHP :

- Affichage de la section entête
- Affichage du logo
- Affichage de la navigation
- Affichage des messages éventuels
- Affichage d'une liste des titres des articles : un clic sur le titre de l'article va rediriger le visiteur sur la page article.php qui affichera l'article en entier. L'ID de l'article sera passé dans l'URL (méthode \$_GET)
- Affichage du footer

Au-dessus de la structure, avant celle-ci, on créera le contrôleur PHP.

Le contrôleur PHP implémentera :

Le require du fichier settings.php

La vérification de l'objet de connexion \$conn

L'appel à la fonction qui va chercher les articles publiés

Le test du retour de la fonction : on vérifie que nous avons reçu un format de données autorisé.

C'est ce retour, ces données qui seront passées à la fonction d'affichage de la liste des articles dans la structure HTML en dessous du contrôleur.

Remarque : Il faut créer une fonction pour gérer les affichages des messages.