Développer un thème avec Wordpress et Docker

Installation et configuration

- Au préalable, installer Git, Docker et Docker-Compose sur votre machine de développement
  - Vous pouvez suivre la procédure d'installation de Docker et Docker-Compose sous Debian 10
- Commencer par cloner en local le dépôt cbn-alpin/sinp-paca-srv : `git clone git@github.com:cbn-alpin/sinp-paca-srv.git`
- En ligne de commande, se placer dans le dossier suivant : `cd sinp-paca-srv/web-srv/docker/cms.silene.eu`
- Créer un fichier *.env* adapté à votre environnement en se basant sur *.env.sample* : `cp .env.sample .env`
  - Utiliser la commande `id` dans un terminal pour afficher vos identifiants numériques correspondant à votre utilisateur (=UID) et groupe (=GID). Renseigner la propriété *WP_DEV_FPM_USER* avec au format *UID:GID*.
  - Si vous n'avez pas la possibilité de créer le lien symbolique comme indiqué ci-dessous, vous pouvez indiquer le chemin vers le dossier local de votre thème en cours de création dans la propriété *WP_DEV_THEME_PATH*.
  - Enfin, la propriété *WP_DEV_THEME_NAME* permet d'indiquer le nom du dossier du thème (par défaut "*silene*") qui sera placé dans le dossier */var/www/html/wp-content/themes/* du container *cms-wordpress*.
- Dans le dossier *wordpress* créer un lien symbolique vers le dossier hébergeant votre thème en cours de création : `ln -s <mon-chemin-vers-theme> wordpress/theme`
- Pour lancer localement en mode développement le CMS Worpdress utiliser la commande : `docker-compose -f docker-compose.yml -f docker-compose.dev.yml up`
  - Le CMS devrait être accessible sur l'adresse locale : http://127.0.0.1:50080
  - Lorsque cela semble fonctionnel vous pouvez ajouter l'option `-d` pour lancer en tant que *daemon* et ne plus voir les logs des services dans le terminal : `docker-compose -f docker-compose.yml -f docker-compose.dev.yml up -d`

Notes sur la gestion des permissions

- Le dossier de travail local du thème est partagé avec 2 containers (*cms-nginx*, *cms-wordpress*). Afin qu'il soit accessible par les utilisateurs de ces containers, il est nécessaire de s'assurer que les dossiers qu'il contient ont bien les droits 755 et les fichiers 644 (au moins durant la phase de développement). Lorsque c'est possible, il est aussi plus pratique que l'utilisateur de l'application du container (PHP-FPM ou Nginx) utilise le même UID et GID que votre utilisateur local.
- Pour le container *cms-wordpress* (basé sur l'image PHP officiel - variante FPM), il est nécessaire de le lancer avec un utilisateur dont l'id est identique à celui de votre utilisateur local (généralement 1000). C'est ce que permet de faire l'attribut "*user:*" du service *cms-wordpress* dans le fichier *docker-compose.dev.yml*. De cette façon, toute modification effectuée sur les fichiers locaux sera répercuté dans le container et restera accessible par l'utilisateur de *PHP-*

FPM. Voir la section "Running as an arbitrary user" dans la documentation concernant l'image Wordpress et la même dans la documentation de l'image PHP.

- Concernant le container *cms-nginx*, le changement d'utilisateur n'étant pas évidant, il suffit de s'assurer pour le dossier de travail local que :
  - les dossiers aient des droits en lecture et exécution pour "les autres" (=5).
  - les fichiers aient des droits en lecture pour "les autres" (=4).

From:
https://sinp-wiki.cbn-alpin.fr/ - **CBNA SINP**

Permanent link:
https://sinp-wiki.cbn-alpin.fr/serveurs/sinp-paca/cms-docker-devel?rev=1582884896

Last update: **2020/02/28 10:14**