Installation d'un environnement de développement local

Ce document explique le fonctionnement d'un ensemble de playbooks Ansible qui installe les sources et configure, pour le développement local, l'ensemble des briques de Publik. Pour mieux comprendre le rôle de toutes ces briques, lisez le document présentation générale.

Pré-requis système

- Debian bookworm (stable) ou trixie (testing) : l'environnement de développement Publik fonctionne sous Debian. Si vous utilisez un autre système nous conseillons vivement d'installer une machine virtuelle ou un conteneur Linux sous Debian.
- L'utilisateur linux utilisé doit être sudoer mais ne doit pas être l'utilisateur **root**.

Installer les logiciels suivant :

- Ansible (une version supérieure ou égale à 2.5 est nécessaire)
- Git
- GNU Make

Via la commande :

```
    sudo apt install git ansible make
```

Installer Publik

Ne pas lancer les commandes en tant qu'utilisateur root

- Clonez le dépôt git publik-devinst :

```
    git clone https://git.entrouvert.org/publik-devinst.git && cd publik-devinst
```

- lancer l'installation de Publik

```
  make install
```

Votre mot de passe sudo vous sera demandé, voir ici si vous voulez savoir pourquoi.

- lancer le déploiement d'un tenant

```
  make deploy
```

Vous avez maintenant une instance de Publik fonctionnelle qui tourne sur votre machine, accessible à cette adresse : https://combo.dev.publik.love/.

Utilisateur : **admin@localhost**
Mot de passe : **admin**

1. Après votre première authentification, si vous ne voyez pas d'entrée "Formulaires" et/ou "Workflows" dans le menu, une visite sur la page suivante devrait résoudre le problème : https://wcs.dev.publik.love/login .
2. Sur les pages "Fabrique de formulaires" vous sera indiqué la nécessité de "définir des **rôles**", rendez vous sur https://authentic.dev.publik.love/manage/roles/ pour créer un rôle.

Administration système

- Le code source des briques de publik sont disponibles dans dépôts git dans /home/utilisateur/src
- Les briques sont installés dans un virtualenv python qui se trouve ici /home/utilisateur/envs/publik-env-py3

Services

Les services qui constituent votre instance de Publik sont les suivants :

- authentic-multitenant

- bijoe
- chrono
- combo
- fargo
- hobo
- hobo-agent
- passerelle
- wcs
- welco

Les services sont démarrés automatiquement et sont gérés par supervisord.
Pour faire des opérations sur un service :

```
sudo supervisorctl {start|status|stop} django:nomduservice
```

Par exemple :

```
sudo supervisorctl {start|status|stop} django:passerelle
```

Les logs des service sont disponibles dans /var/log/nomduservice/stderr.log et /var/log/nomduservice/stdout.log

Pour faire tourner un service dans une console :

```
sudo supervisorctl stop django:nomduservice
/home/utilisateur/envs/publik-env-py3/bin/nomduservice-server
```

Par exemple :

```
sudo supervisorctl stop django:passerelle
/home/utilisateur/envs/publik-env-py3/bin/passerelle-server
```

Des directives de configuration django peuvent être ajoutés dans un ou plusieurs fichiers python, à placer dans /home/utilisateur/.config/publik/settings/nomduservice/settings.d/.
En revanche ne pas modifier /home/utilisateur/.config/publik/settings/nomduservice/settings.py, fichier de configuration principal d'un service, qui sera écrasé lors d'une mise à jour de votre installation.

Utilisation du serveur web UWSGI

Les services lancés par défaut utilisent le serveur de développement django. Il est possible d'utiliser un "vrai" serveur web à la place, uwsgi. D'abord couper le service de base, puis lancer le service uwsgi :

```
sudo supervisorctl stop django:nomduservice
sudo supervisorctl start uwsgi:nomduservice-uwsgi
```

Par exemple :

```
sudo supervisorctl stop django:passerelle
sudo supervisorctl stop uwsgi:passerelle-uwsgi
```

Pour visualiser la journalisation de UWSGI :

```
sudo supervisorctl tail -f  uwsgi:nomduservice-uwsgi stderr
```

Pour un service donné le fichier de configuration uwsgi se trouve dans /home/utilisateur/.config/publik/settings/nomduservice/uwsgi.ini.
La personnalisation de cette configuration peut être faite dans un fichier /home/utilisateur/.config/publik/settings/nomduservice/setttings.d/uwsgi-local.ini.

Commandes d'administration

Les commandes d'administration de chaque service sont disponibles dans /home/utilisateur/envs/publik-env-py3/bin/nomduservice-manage, par exemple :

```
/home/utilisateur/envs/publik-env-py3/bin/hobo-manage
```

Mettre à jour son installation

D'abord mettre à jour l'utilitaire d'installation :

```
    cd ~/src/publik-devinst && git pull
```

Mise à jour complète

Via la même commande que lors de l'installation.

```
  make install
```

Attention si vous avez fait des commits dans la branche main d'une des briques de Publik et que ces commits ne sont pas poussés sur nos dépôts, ces commits seront perdus.
En revanche si vous avez travaillé dans vos propres branches, tout va bien se passer.

Cette opération pouvant prendre un certain temps, nous proposons des raccourcis plus rapides si vous désirez effectuer uniquement les taches suivantes : mise à jour du certificat TLS ou mise jour du code source.

Mises à jour ciblées

Les mise à jour ciblées décrites ci-dessous sont déjà effectuées lors d'une mise à jour complète. Une mise à jour complète pouvant prendre un temps conséquent, nous proposons quelques raccourcis.

Mise à jour du certificat TLS

Nous proposons un certificat wildcard qui couvre tous les noms *.dev.publik.love, ce certificat est téléchargé à l'installation.
Notez qu'il s'agit d'un certificat LetsEncrypt, donc sa durée de vie n'est pas longue, trois mois au mieux. Si la version que vous aviez téléchargé a expiré,
vous pouvez télécharger un certificat valide avec la commande :

```
make renew-certificate
```

Mise à jour du code source

Seront également effectuées quelques opérations afférentes à la mise jour du code source.

```
make upgrade
```

Utilisation avancée

Personnalisation de l'installation

Le comportement de l'installation peut être personnalisé :

- en copiant le fichier local-inventory.yml.example vers un fichier local-inventory.yml
- puis en éditant local-inventory.yml et en modifiant les variables de configuration qui s'y trouvent

Les variables les plus communément utilisées sont les suivantes :

- clean_venv (true/false) : suppression de l'environnement virtuel python existant avant l'installation (par défaut false)
- compile_theme (true/false) : compilation des thèmes graphiques (par défaut true)
- delete_all_tenants (true/false) : lors de la suppression des tenants, suppression de tous les tenants (par défaut true, si false, suppression des tenants déclarés uniquement)
- git_ssh (true/false) : cloner les dépôts git en utilisant le protocole ssh au lieu de https (par défaut false)
- src_dir : le répertoire dans lequel les dépots git seront clonés (par défaut /home/nom-de-votre-utilisateur/src)

Déploiement d'un nouveau tenant

D'abord, notez que chaque brique de Publik tourne dans un serveur web sécurisé indépendant et fonctionne selon une architecture multi-tenants.
Chaque tenant (une instance locale d'une brique Publik) est identifié par son URL.
Vous pouvez déployer de nouveaux tenants en plus de ceux déployés par défaut.
Pour ce faire, dans le fichier local-inventory.yml décrit précédemment, dé-commenter les lignes de la section tenants_conf et remplacer customname par un nom de votre choix,
puis invoquer la commande de déploiement de tenant.

```
make deploy
```

(qu'il faut parfois lancer plusieurs fois et être patient en raison des messages RabbitMQ qu ne sont pas consommés ni ackés à cette date)

Pendant l'exécution, vous pouvez lire les logs de votre "agent" hobo qui configure vos tenants :

```
sudo tail -f /var/log/hobo-agent/stderr.log
```

Suppression des tenants

Par défaut tous les tenants seront supprimés

```
make delete
```

Pour supprimer uniquement certains tenants, il faut que l'option delete_all_tenants soit mise à false. Seront alors supprimés uniquement les tenants déclarés dans la section tenants_conf de votre configuration.

Suppression complète de l'environnement de développement

```
make clean
```

DNS

La zone dev.publik.love est configurée pour que tous les noms foobar.dev.publik.love aient comme adresse 127.0.0.1, c'est-à-dire votre machine.

Pour vérifier que la résolution DNS fonctionne :

```
$ host foobar.dev.publik.love
foobar.dev.publik.love has address 127.0.0.1
```

Si le serveur DNS de votre fournisseur d'accès est menteur (par exemple chez l'opérateur Free) alors ça ne marchera pas. Utilisez plutôt un service de DNS correct tel que Quad9 (9.9.9.9) ou installez un résolveur chez vous.

Si vous travaillez sans connexion Internet, ou si vous utilisez un autre nom de domaine, vous pouvez éditer votre fichier /etc/hosts, par exemple :

```
# à poser à la fin de /etc/hosts, à adapter selon les noms des sites que vous avez choisi de déplo
yer
127.0.0.1  agent-combo.dev.publik.love authentic.dev.publik.love bijoe.dev.publik.love chrono.dev.
publik.love combo.dev.publik.love fargo.dev.publik.love hobo.dev.publik.love  passerelle.dev.publi
k.love wcs.dev.publik.love
```

Mot de passe sudo

Une série de fichiers à l'extérieur du répertoire personnel doivent être modifiés, c'est la raison pour laquelle l'accès "sudo" est demandé lors de l'installation.
Voici le liste des actions effectuées par le playbook qui nécessitent les droits administrateurs :

- installation de paquets systèmes (la liste est disponible au début du fichier roles/base/tasks/main.yml)
- création d'un rôle et de bases de données postgres
- création de fichiers de configuration supervisord dans /etc/supervisor/conf.d
- création de fichiers de configuration nginx dans /etc/nginx/sites-enabled et /etc/nginx/sites-available
- création d'une clé secrète pour chaque service de publik (/etc/nomduservice/secret)
- création d'un répertoire dans /var/lib/nomduservice pour chaque application
- création d'un répertoire de log dans /var/log/nomduservice pour chaque application

Obtenir de l'aide

En cas de problème, vous pouvez obtenir de l'aide en créant une demande de support ici :
https://dev.entrouvert.org/projects/publik-devinst/issues, merci d'y décrire aussi précisément que possible le problème rencontré.

Si l'installation échoue, joindre à votre demande un fichier de log de l'installation lancée en mode verbeuse.

Un tel fichier peut par exemple s'intituler debug.log et s'obtenir en lançant l'installation de la façon suivante :

```
make install > debug.log
```

Si l'installation déroule jusqu'à la fin mais que l'instance obtenue vous semble non fonctionnelle, merci de joindre des captures d'écran, etc.