Hobo - Development #21723

Permettre aux applications "tenantisées" de modifier les settings ETC_DIR, VAR_DIR et des variables qui en découlent

08 février 2018 11:51 - Emmanuel Cazenave

Statut:	Rejeté	**Début:**	08 février 2018
Priorité:	Normal	**Echéance:**	
Assigné à:	Emmanuel Cazenave	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	

Description

Dans debian_config_common.py, on définit VAR_DIR et ETC_DIR comme suit : ETC_DIR = '/etc/%s' % PROJECT_NAME, VAR_DIR = '/var/lib/%s' % PROJECT_NAME.

D'autres variables étant calculées à partir des ces deux là dans debian_config_common.py, il n'est pas aisé pour une application "tenantisée", utilisant le traditionnel execfile('/usr/lib/hobo/debian_config_common.py') de changer les valeurs de ETC_DIR, VAR_DIR et des variables qui en découlent.

Idée de Fred :

```
ETC_DIR = os.path.join(ETC_BASE_DIR if 'ETC_BASE_DIR' in globals() else '/etc/', PROJECT_NAME)
```

Demandes liées:

Lié à Publik Installation Développeur - Development #21643: Rendre configurab...	**Nouveau**	**05 février 2018**
Lié à Hobo - Development #22892: Conflit tests unitaires et installation mult...	**Fermé**	**29 mars 2018**

Historique

#1 - 08 février 2018 11:55 - Emmanuel Cazenave

- Lié à Development #21643: Rendre configurable le répertoire des tenants ajouté

#2 - 08 février 2018 12:01 - Emmanuel Cazenave

- Fichier 0001-ease-customization-of-VAR_DIR-and-ETC_DIR-settings-v.patch ajouté

- Statut changé de Nouveau à En cours

- Assigné à mis à Emmanuel Cazenave

- Patch proposed changé de Non à Oui

J'utilise ici locals() au lieu de globals() ce qui rend la chose plus facile à tester unitairement et garde le comportement attendu (en plus des tests unitaires, j'ai vérifié que les variables définies dans un module sont bien dans le locals() d'un deuxième si le premier execfile le deuxième).

#3 - 08 février 2018 13:40 - Thomas Noël

J'arrive après la bataille, mais on peut pas plutôt partir sur des variables d'environnement ? Je suis pas super à l'aise avec ces "locals", je vois pas d'où elles pourraient venir, le "debian_config_common.py" il est execfilé super tôt.

Quelque chose genre :

```
ETC_PREFIX = os.getenv('%s_ETC_DIR' % PROJECT_NAME.upper()) or '/etc'
```

#4 - 08 février 2018 13:56 - Emmanuel Cazenave

- Fichier 0001-ease-customization-of-VAR_DIR-and-ETC_DIR-settings-v.patch ajouté

Oui après réflexion, ça craint en fait mes histoires de locals, nouveau patch avec les globals comme d'habitude, j'ai adapté les tests.

Je ne lis ton message que maintenant : je préférerais ne pas rajouter de variables d'environnement, je trouve ça déjà assez difficile à suivre ce qui se passe avec les variables d'env APPS_SETTINGS, les lectures dynamiques de settings par d'autres settings, les petits "et pour finir je vais regarder dans /etc/app si il y a pas un petit settings qui reste à charger...

#5 - 08 février 2018 15:21 - Benjamin Dauvergne

Je rebondis sur l'autre ticket et celui-ci, ne serait-ce pas plus simple comme pour les Makefile qui doivent pouvoir s'installer sur un système mais aussi dans un répertoire temporaire pour du packaging (dpkg, rpm, etc..) d'avoir juste une variable d'environnement DESTDIR ? Par défaut on met '/', mais si on '/opt/publik/' ça suppose que tou y est. Ça n'empêche pas de pouvoir personnaliser /etc, /var mais pour une installation locale ou contenairisé ça me parait plus pratique.

#6 - 08 février 2018 15:41 - Thomas Noël

Benjamin Dauvergne a écrit :

> (...) une variable d'environnement DESTDIR

Je pousserais un peu plus loin avec <PROJECT_NAME>_DESTDIR, histoire de, et en variable d'env ça serait bien, je suis d'accord.

Emmanuel, quand au fait de fixer une variable d'environnement, ça peut être fait simplement lors du lancement du uwsgi ou du runserver, ça me parait pas mortel du tout... non ? On a bien déjà une variable pour <PROJECT_NAME>_SETTINGS_FILE (cf par exemple la fin de combo/settings.py) ; on serait donc exactement sur le même principe.

#7 - 08 février 2018 15:51 - Emmanuel Cazenave

> Je pousserais un peu plus loin avec <PROJECT_NAME>_DESTDIR, histoire de, et en variable d'env ça serait bien, je suis d'accord.

Pas sûr de comprendre : si cette variable d'environnement est définie, toutes les autres variables de type 'filesystem' deviennent des sous chemin c'est ça l'idée ?

#8 - 08 février 2018 16:00 - Benjamin Dauvergne

Oui il faut réécrire tous les autres chemins en XXX_DIR = os.path.join(DESTDIR, XXX_BASE_DIR, PROJECT_NAME).

#9 - 08 février 2018 16:47 - Frédéric Péters

Pour rappel du contexte, c'est l'exploitation dans un environnement de dev local des debian_config*; moi ça m'irait bien qu'on ne chamboule pas tout pour cet usage, c'est pour ça que je suggérais la ligne citée en description, en tapant un commentaire au-dessus, genre, "allow a local development environment to override default directory". Et ma préférence allait à la modification des deux variables existantes pour ne pas devoir également aller modifier d'autres projets (genre HOBO_SKELETONS_DIR = os.path.join(VAR_DIR, 'skeletons') dans authentic).

#10 - 08 février 2018 16:53 - Benjamin Dauvergne

Ok ça me va qu'on ne supporte cela que pour ETC_DIR et VAR_DIR, mais je continue à penser qu'un simple BASEDIR/DESTDIR dans une variable d'environnement et le plus simple (peut-être lui donné un nom plus compliqué pour éviter une collision). je ne vois pas l'intérêt de dire à combo qu'il va à un endroit et que chrono irait ailleurs.

#11 - 08 février 2018 17:25 - Emmanuel Cazenave

Je comprends l'idée du DESTDIR, mais j'avoue que je m'en passerai bien parce que ça parait moins maitrisable que mon patch (vous allez voir avec ce qui suit), pour le fait que ce soit une varibale d'environement je suis plus que sceptique.

Pardon pour le pavé qui suit, mais j'élargis un peu la discussion.

Je reprends le workflow de chargement des settings de combo en mode mutlitenant (j'ai surement pas tout compris, vous me corrigerez), dans le cas de la commande manage :

- la commande combo-manage charge le fichier classique combo.settings (parce qu'elle positionne DJANGO_SETTINGS_MODULE à combo.settings dans os.environ)
- combo.settings charge /usr/lib/debian_config.py parce que COMBO_SETTINGS_MODULE est positionné dessus dans os.environ (j'ai pas compris qui/quoi réglait COMBO_SETTINGS_MODULE dans le cas d'une install debian)
- /usr/lib/debian_config.py charge '/usr/lib/hobo/debian_config_common.py'
- /usr/lib/debian_config.py charge un ETC_DIR/settings.py ('ETC_DIR' ayant été définis par '/usr/lib/hobo/debian_config_common.py' à /etc/combo)
- /usr/lib/debian_config.py charge '/usr/lib/hobo/debian_config_settings_d.py'
- '/usr/lib/hobo/debian_config_settings_d.py' charge tout les fichiers.py qui se trouvent dans /etc/combo/settings.d/

Les gars excusez moi, mais j'appelle ça L'ENFERT SUR TERRE DES SETTINGS DE PUBLIK.

Je rêve déjà du jour où il y aura un et un seul fichier de configuration pour une application (positionné par le DJANGO_SETTINGS_MODULE des familles), que tu peux aller lire tranquillement pour savoir quels sont les settings sur le déploiement particulier de l'application. Et où la redondance

entre les fichiers de configuration des applications de publik est gérée en amont par un outil de déploiement (avec du templating de fichiers de conf peut-être), putôt que par du chargement dynamique de settings (merci django qui permet de faire des trucs pareil).

Donc rajouter une variable d'environnement en plus là dedans, je crois que je vais vraiment avoir du mal, j'aurais l'impression de me taper moi même sur la tête.

Voilà pardon si c'est un peu frontal, mais il fallait que je le dise.

#12 - 08 février 2018 17:32 - Benjamin Dauvergne

T'as oublié les settings_loaders qui chargent /var/lib/<project>/tenants/<hostname>/settings.{json,py} ;)

#13 - 08 février 2018 17:36 - Benjamin Dauvergne

Mais je ne vois toujours pas ce qu'avoir une variable d'environnement PUBLIK_BASEDIR aurait de gênant; tout ça c'est parce qu'on veut mutualiser un max de bouts des settings.py sans copier/coller tout dans chaque projet et laisser un max de possibilité de customisation soit par brique soit par tenant et aussi pouvoir poser des la config via puppet sans se marcher sur les pieds (d'où les nouveaux /etc/<projects>/settings.d/*.py).

Si tu proposes un truc plus simple tout en respectant ces contraintes, be my guest !

#14 - 08 février 2018 17:45 - Benjamin Dauvergne

Mais je reconnais qu'il y a peut-être un peu trop de politique dans les settings des paquets en eux même, mais ça vient du fait qu'on ne savait pas vraiment gérer de la configuration à l'époque alors on a préféré tout mettre dans les paquets; si on faisait les choses aujourd'hui probablement qu'on ne mettrait quasiment rien dans le paquet et tout dans de la config posée par puppet.

#15 - 08 février 2018 17:52 - Emmanuel Cazenave

Ce qui me gêne avec la variable d'environnement c'est que ça complique la compréhension des settings.
Là on s'en sort en lisant tous les fichiers de conf, ce qu'il faut faire quoiqu'il arrive.
Variable d'environnement il faut en plus aller voir ailleurs qui/quoi l'a positionné avant de lancer la commande/démon, enfin bon c'est sur que c'est pas insurmontable mais c'est surtout que j'ai l'impression que ça va dans le sens complexification plutôt que simplification.

#16 - 08 février 2018 17:55 - Frédéric Péters

> Les gars excusez moi, mais j'appelle ça L'ENFER SUR TERRE DES SETTINGS DE PUBLIK.

Et bientôt, l'enfer des templates de Publik…

Mais beaucoup de choses ont une bonne raison d'être là, le gros problème c'est qu'on ne devrait pas demander à chacun de suivre les fils, il y avait un début, il doit être continué, sur la page https://dev.entrouvert.org/projects/prod-eo/wiki/HowDoWeDoDjangoSettings

(concernant les variables d'environnement, de mon côté je ne suis pas fan non plus, on en a (ab)usé à un moment, ça me va plutôt bien qu'on ait arrêté).

#17 - 08 février 2018 18:18 - Emmanuel Cazenave

Frédéric Péters a écrit :

> Mais beaucoup de choses ont une bonne raison d'être là, le gros problème c'est qu'on ne devrait pas demander à chacun de suivre les fils, il y avait un début, il doit être continué, sur la page https://dev.entrouvert.org/projects/prod-eo/wiki/HowDoWeDoDjangoSettings

Je vais faire ça, puisque je suis le dernier en date à à voir tiré les fils.

#18 - 08 février 2018 18:38 - Benjamin Dauvergne

Alors pour moi cette variable d'environnement c'est juste pour permettre une installation hors paquet Debian du bouzin sans devoir écrire des settings spécifiques, et donc ça me parait raisonnable, ça ne doit jamais être utilisé en condition normale (prod/Debian), je ne vois vraiment aucun autre moyen d'appliquer une modification à toutes les briques sans devoir modifier n fichiers ni prévoir un emplacement global où serait cette information (genre un /etc/publik/settings.d/ global, mais on revient au problème du début, comment on fait pour déplacer cet emplacement).

#19 - 08 février 2018 19:23 - Emmanuel Cazenave

Si je reviens à mon problème de départ #21643 je suis quoi qu'il arrive obligé d'écrire des settings particuliers par application (ce qui ne me choque pas pour le coup, à un déploiement particulier des settings particuliers).

La trame générale est là http://git.entrouvert.org/publik-devinst.git/tree/roles/app-setup/templates/app-settings-base.j2, avec de l'héritage pour tenir compte des spécifités de chaque application.

Je désactive des loggers handler par exemple qui vont m'embêter dans un environment de dev, je bidouille pour que le request du virtualenv fasse pas des erreurs SSL. Enfin j'arrive à faire ma vie avec un enchaînement de loading de settings à 3 niveaux : 'app.settings' qui charge mes settings

spécifiques pointé par 'APP_SETTINGS_FILE' qui charge 'debian_config_common' (celui que je trouve dans {{src_dir}/hobo/debian/, et je mets rien dans les /etc/app). Je pourrais m'en sortir sans ce patch, mais le patch simplifie un peu mes settings spécifiques.
Je vois pas comment je pourrais faire sans un fichier de settings spécifiques et ce n'était pas mon but.

#20 - 09 février 2018 10:38 - Emmanuel Cazenave

Un avec un peu de recul, excusez moi pour "L'ENFERT....".
J'en ai bavé dans publik-devinst, la cocotte minute a explosé ici, à retardement, désolé.

#21 - 09 février 2018 11:29 - Benjamin Dauvergne

Je trouvais bien de passer exactement par les mêmes fichiers que ceux du packaging (quitte à se contorsionner un peu), s'il faut désactiver des handlers que ce soit possible et pour le SSL et bien j'aurai préféré que ça marche tout seul avec du SSL. L'idée étant que si on utilise un telle plate-forme de dév pour attaquer un web-service chez un éditeur qui fait du SSL sans certificat valide, ça se voit par exemple. Chaque fois qu'un environnement de dév diffère de la prod, ça finit par nous rattraper.

#22 - 09 février 2018 14:16 - Emmanuel Cazenave

J'ai essayé de rapprocher au maximum de la prod.
Enfin je charge le debian_config_common, mais pas les debian_config tout court, j'y ai pas pensé, je pourrais essayer de regarder ça.
D'accord pour SSL et je ne le désactive pour request, je fais juste en sorte qu'il se comporte comme le python-request système (http://git.entrouvert.org/publik-devinst.git/tree/roles/app-setup/templates/app-settings-base.j2#n13). Elias s'est pris des erreurs SSL parce qu'il avait mal configuré ses certificats, c'est pas open bar.
Si je fais pas ça, je me retrouve ici #21644, où c'est bien compliqué.

#23 - 29 mars 2018 15:03 - Emmanuel Cazenave

- Lié à Development #22892: Conflit tests unitaires et installation multitenant ajouté

#24 - 04 mars 2019 11:25 - Benjamin Dauvergne

- Statut changé de En cours à Solution proposée

Je passe en solution proposée, si c'est mort, le dire.

#25 - 04 mars 2019 14:16 - Emmanuel Cazenave

- Statut changé de Solution proposée à Rejeté

Devinst marche bien, oublions.

Fichiers

0001-ease-customization-of-VAR_DIR-and-ETC_DIR-settings-v.patch	2,92 ko	08 février 2018	Emmanuel Cazenave
0001-ease-customization-of-VAR_DIR-and-ETC_DIR-settings-v.patch	3,48 ko	08 février 2018	Emmanuel Cazenave