TD d'intégration continue avec Travis-CI

Cécilia Ostertag, Franck Soubes, Guillamaury Debras, Peter Bock

5 février 2018

Table des matières

1 But du TD

Ce TD vise a expliquer l'intérêt de l'Intégration Continue, en utilisant Travis-CI lié à un dépôt Github pour démontrer comment on peut exécuter un ensemble de tests sur ses fonctions, à chaque commit, pour être sûr que les commits ne cassent pas la fonctionnalité du script.

2 Création de compte et intégration avec Github

La première étape est de lier son compte github avec TRAVIS-CI, l'outil d'intégration continue que nous utiliserons pour ce TD, sur le site de Travis-CI .

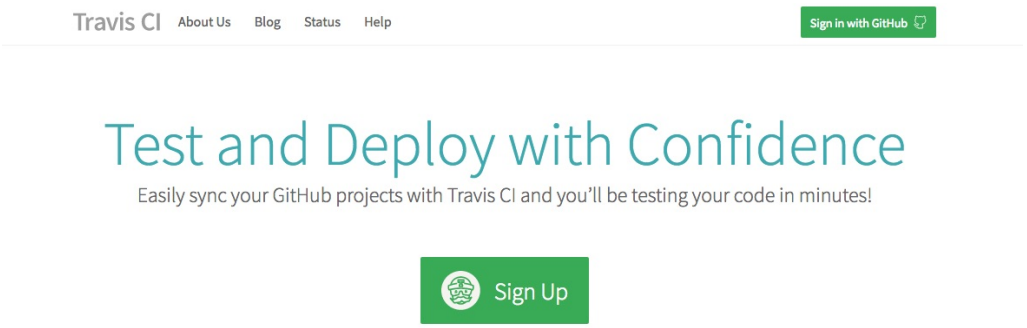

FIGURE 1 – le grand boutton vert, vous pouvez pas le rater.

2.1 forker le dépôt du TD et l'activer dans Travis-CI

Nous avons préparé un Dépôt Github contenant 2 scripts Python, ainsi que 2 tests sur ces scripts, ici.

Quand vous l'aurez forké dans votre propre compte Github, vous pourrez l'activer dans Travis à travers votre profil Travis-CI

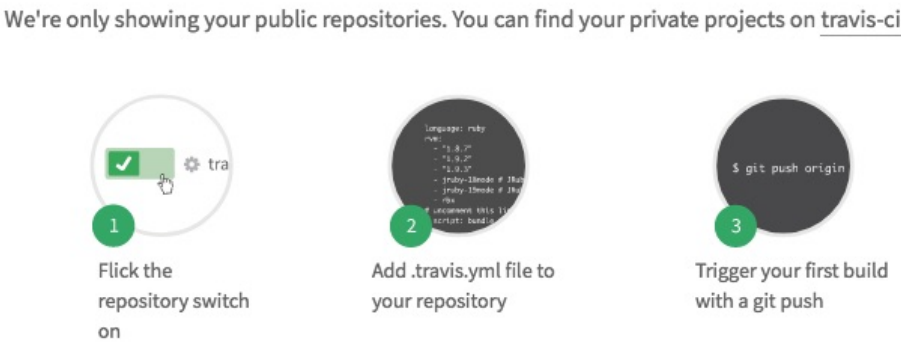

FIGURE 2 –

3 Création du .travis.yml

Pytest est l'un des modules de tests pré-intégré dans l'environnement de test par défaut dans Travis.

Cet environnement peut être adapté selon les besoins de l'utilisateur, ceci est fait en manipulant le fichier .travis.yml, qui contient l'entiereté de la configuration de l'environnement de test, ainsi que les commandes à executer pour tester le dépôt github dans lequel il se trouve.

Créez un fihcier *.travis.yml* dans votre dépôt, et mettez y l'information suivante :

```
language: python
python:
      - "2.6"
      - "3.6"
script:
        - pytest
```

Ici, le fichier .travis.yml dit que le langage du dépôt est le python, de faire tous les tests en utilisant les version de python citée (2.6 et 3.6).

Le tag Script détermine les programmes a exécuter.

Pytest est un programme qui exécute tous les test contenus dans des fichiers dont le nom correspond a la syntaxe *test_ *.py* ou **_ test.py*.

Pour plus d'informations sur comment adapter le fichier .travis.yml, dont l'adaptation de l'environnement de test, l'installation des dépendances, etc., allez ici.

4 Résultats du premier build et troubleshooting

Après le premier build, vous devriez avoir des résultats comme suivent :

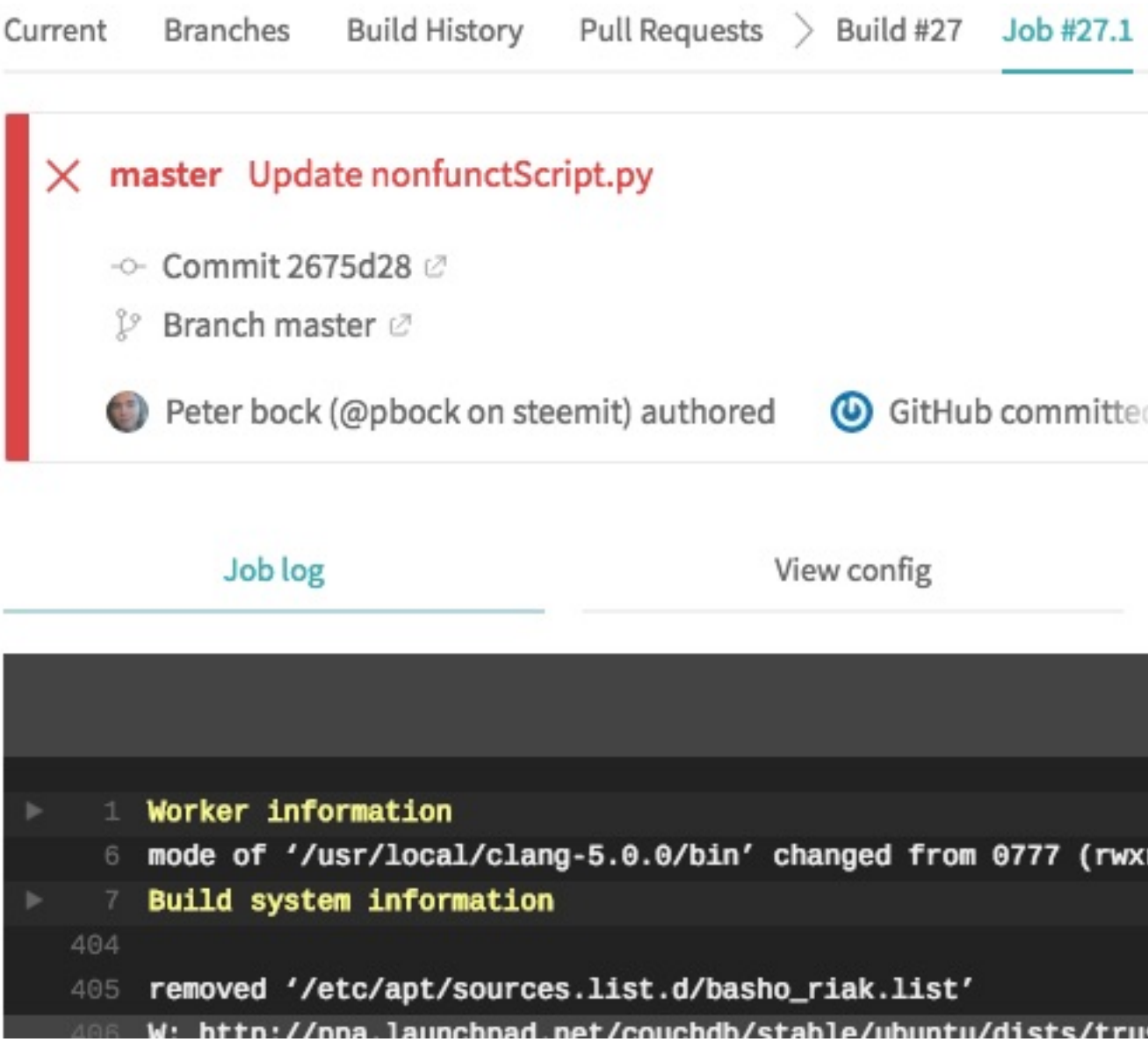
Current
Branches
Build History
Pull Requests
Build #27
Job #27.1
master Update nonfunctScript.py
Commit 2675d28
Branch master
Peter bock (@pbock on steemit) authored
GitHub committe
Job log
View config
1 Worker information
6 mode of '/usr/local/clang-5.0.0/bin' changed from 0777 (rwx
7 Build system information
404
405 removed '/etc/apt/sources.list.d/basho_riak.list'
406 W: http://ppa.launchpad.net/couchdb/stable/ubuntu/dists/tru

FIGURE 3 –

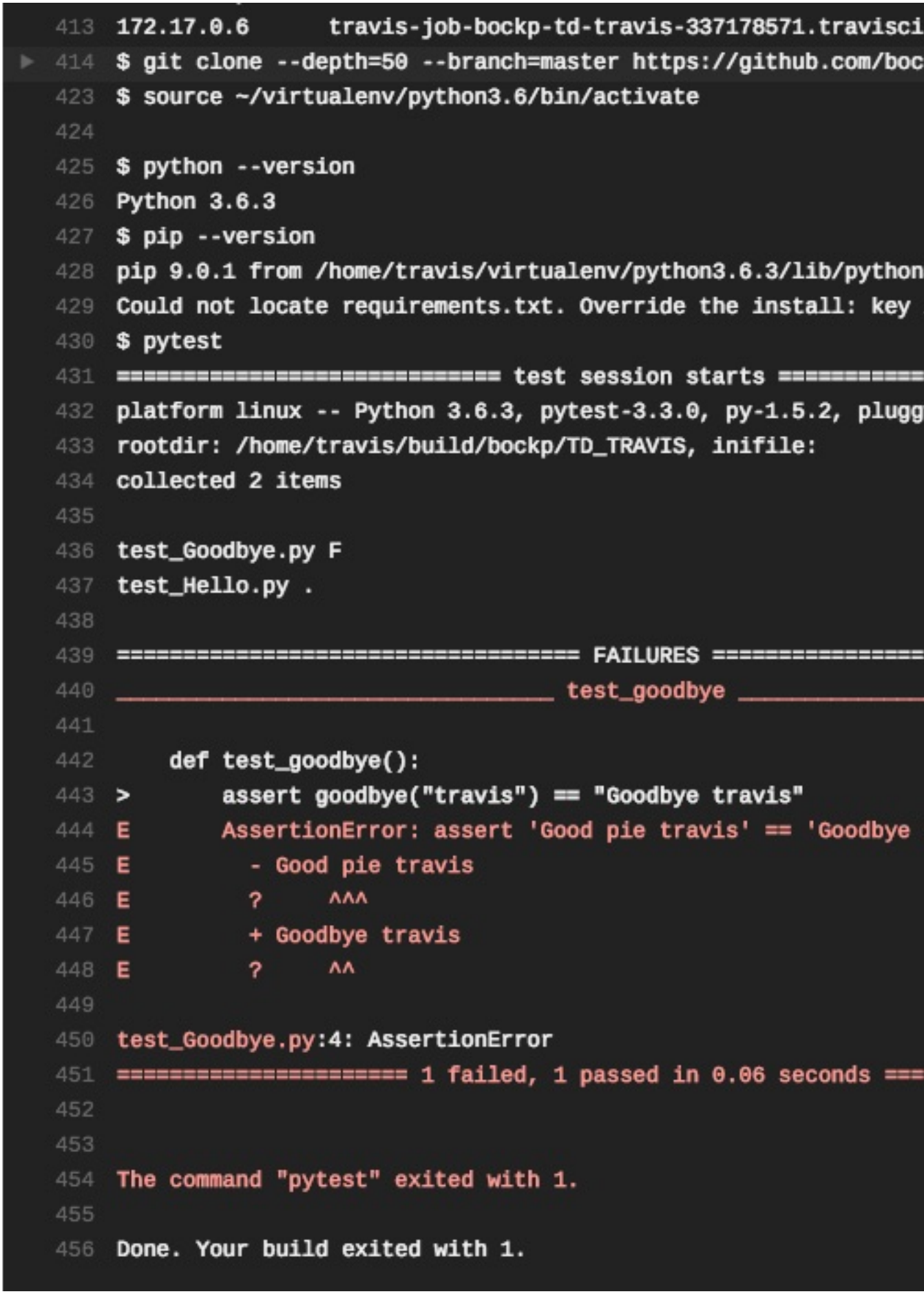
413 172.17.0.6 travis-job-bockp-td-travis-337178571.travisci
414 $ git clone --depth=50 --branch=master https://github.com/boc
423 $ source ~/virtualenv/python3.6/bin/activate
424
425 $ python --version
426 Python 3.6.3
427 $ pip --version
428 pip 9.0.1 from /home/travis/virtualenv/python3.6.3/lib/python
429 Could not locate requirements.txt. Override the install: key
430 $ pytest
431 ============================= test session starts ============
432 platform linux -- Python 3.6.3, pytest-3.3.0, py-1.5.2, plugg
433 rootdir: /home/travis/build/bockp/TD_TRAVIS, inifile:
434 collected 2 items
435
436 test_Goodbye.py F
437 test_Hello.py .
438
439 =================================== FAILURES ===================
440 _____________________________________ test_goodbye _____________
441
442 def test_goodbye():
443 > assert goodbye("travis") == "Goodbye travis"
444 E AssertionError: assert 'Good pie travis' == 'Goodbye
445 E - Good pie travis
446 E ? ^^^
447 E + Goodbye travis
448 E ? ^^
449
450 test_Goodbye.py:4: AssertionError
451 ======================== 1 failed, 1 passed in 0.06 seconds ===
452
453
454 The command "pytest" exited with 1.
455
456 Done. Your build exited with 1.

FIGURE 4 –

Comme vous pouvez voir, une *"erreur"* s'est glissée dans la fonction *goodbye()*, ce qui cause un échec du test qui lui est associé.

Maintenant, votre premier challenge.

Saurez vous corriger cette erreur ?

5 Visualiser l'état d'un build

Pour voir quel est l'état d'un build sans aller sur le site de Travis, vous pouvez ajouter l'icone suivant dans votre Readme :

FIGURE 5 –

voici le code a mettre pour avoir l'icône dans le readme, en remplaçant NOM_GITHUB par votre nom d'utilisateur de github, et NOM_DEPOT par votre nom de dépôt.

```
[![Build status](https://travis-ci.org/<NOM_GITHUB>/<NOM_DEPOT>.svg?master)](https://t
```

6 A vous de jouer

Quand vous aurez fait le commit corrigeant l'erreur dans goodbye, et vérifié que tout fonctionne bien, a votre tour de jouer.

Créez une fonction, et son test associé, et rajoutez les à votre dépôt.

Vous connaissez maintenant les fonctionnalités de base de Travis-CI.

Pour en savoir plus, consulter la documentation de l'outil.