PYTHON*

Менеджер версий python

ИЗ ПЕСОЧНИЦЫ

prokoptsev 25 ноября 2013 в 12:41 47,3k

Иногда полезно держать несколько версий python на одной машине. Допустим для разработки двух проектов нам необходима вторая и третья ветка python. Или вы поддерживаете проект который использует старую версию python.

Обычно для этого мы используем виртуальное окружение virtualenv или же обертку для него virtualenvwrapper. Об этом я рассказывать не буду, так как есть уже много подобных статей, да и в документациях к самим утилитам все очень хорошо объяснено. Достаточно только забить *virtualenv* или *virtualenvwrapper* в поисковик.
Но в дополнение к ним я хочу рассказать в этой статье про менеджер версий python. Кому любопытно прошу под кат.

Чтобы использовать несколько версий python, можно установить их вручную или воспользоваться менеджер версий. Таких есть два: pythonbrew(который более не развивается) и pyenv. Оба менеджера не поддерживают windows(pythonbrew, pyenv) так что питонистам пишущим на этой платформе, придется пока разруливать все руками, либо сделать свою утилиту для смены путей до нужных версий. Кто как справляется с данной ситуацией можете оставлять в комментариях.

Так как *pythonbrew* более не поддерживается в этой статье он рассмотрен не будет.

P.S. В статье приведены примеры проверенные для OS Ubuntu 12.04. При попытке повторить их, делайте поправки относительно своего дистрибутива.

Ручной способ

Для того чтобы работать с несколькими версиями питона, можно установить необходимые версии в указанный префикс. Например чтобы не мудрить с правами, установим дополнительно 2 версии python(2.7.6 и 3.3.2) в директорию пользователю:
2.7.6

```
$ mkdir -p ~/python/src/ && cd ~/python/src/
$ wget http://www.python.org/ftp/python/2.7.6/Python-2.7.6.tar.xz
$ tar -xf ~/python/src/Python-2.7.6.tar.xz && cd ./Python-2.7.6
$ ./configure --prefix=$HOME/python/2.7.6/
$ make && make install
```

для **3.3.2** делаем аналогичные операции:

```
$ wget http://www.python.org/ftp/python/3.3.2/Python-3.3.2.tar.xz
~/python/src/
$ tar -xf ~/python/src/Python-3.3.2.tar.xz && cd ./Python-3.3.2
$ ./configure --prefix=$HOME/python/3.3.2/
$ make && make install
```

Теперь можно создать виртуальное окружение чтобы использовать эти версии:

```
$ virtualenv -p ~/python/2.7.6/bin/python env && .
./env/bin/activate
```

или через virtualenvwrapper:

```
$ mkvirtualenv -p ~/python/2.7.6/bin/python evnwrapper
```

Собственно на основании такого способа описана статья по созданию мультихостинга.
Далее если вам необходимо использовать какую-то из этих версий как python по умолчанию, то вам необходимо добавить в переменную окружения путь до интерпретатора python.

```
$ echo 'export PATH=~/python/2.7.6/bin/' >> ~/.bashrc
```

Соответственно вместо bashrc вы ставите bash_profile, zshrc, kshrc, profile в зависимости от вашей командной оболочки.

```
$ . ~/.bashrc
```

И по необходимости можно установить pip, предварительно установив setuptools.

```
$ wget
https://bitbucket.org/pypa/setuptools/raw/bootstrap/ez_setup.py -O -
| python
$ wget https://raw.github.com/pypa/pip/master/contrib/get-pip.py -O
- | python
```

Фух, ну вроде бы все. А теперь о том как можно сделать это проще используя менеджер версий python.

PyEnv

В общем если вы достаточно ленивы, то можно не делать всего

того что описано выше а воспользоваться утилитой pyenv, которая упростит вам данное взаимодействие с окружением и путями.

Так в чем же заключается особенность этой утилиты? Вот что она может со слов автора проекта:

- Let you **change the global Python version** on a per-user basis.
- Provide support for **per-project Python versions**.
- Allow you to **override the Python version** with an environment variable.
- Search commands from **multiple versions of Python at a time**. This may be helpful to test across Python versions with tox.

По умолчанию все версии Python будут доступны в `~/.pyenv/versions/`. Изменять версии Python можно как в глобальном контексте так и в локальном(например под конкретный проект).

Как ставить pyenv хорошо описывается в инструкции. Так же у автора есть скрипт который по мимо самой pyenv ставит еще и дополнительные плагины, в том числе и для virtualenv. Есть возможность установить плагин и для virtualenvwrapper.

Перед установкой необходимо поставить некоторые зависимости:

```
# apt-get install make libssl-dev zlib1g-dev libbz2-dev libreadline-dev libsqlite3-dev
```

Прежде чем начать установку, убедитесь, что у вас установлен git:

```
# apt-get install git
```

Далее устанавливаем по инструкции:

```
$ git clone git://github.com/yyuu/pyenv.git ~/.pyenv
```

Или так:

```
$ curl https://raw.github.com/yyuu/pyenv-installer/master/bin/pyenv-
installer | bash
```

Во втором случае установка произойдет с дополнительными плагинами.
Далее, для того чтобы все заработало, дополним наш bashrc и перезагрузим оболочку:

```
$ echo 'export PYENV_ROOT="$HOME/.pyenv"' >> ~/.bashrc
$ echo 'export PATH="$PYENV_ROOT/bin:$PATH"' >> ~/.bashrc
$ echo 'eval "$(pyenv init -)"' >> ~/.bashrc
$ . ~/.bashrc
```

Для обновления утилиты или смены ее версии используем git.
Инструкция

Пример использования

```
~ $ pyenv install 2.7.5
~ $ pyenv install 3.3.2
~ $ pyenv rehash
~ $ pyenv versions
* system
  2.7.5
  3.3.2
~ $ pyenv global 2.7.5
~ $ python --version
Python 2.7.5
~ $ cd projects/
~/projects $ pyenv local 3.3.2
```

```
~/projects $ python --version
Python 3.3.2
~/projects $ cd test_prj/
~/projects/test_prj $ python --version
Python 3.3.2
~/projects/test_prj $ cd ..
~/projects $ pyenv local --unset
~/projects $ python --version
Python 2.7.5
```

В добавок ко всему все довольно подробно и детально расписано у автора проекта в его репозиториях на github.

Виртуальное окружение

Все, а дальше как хотите. Если вы используете 3 ветку python то для создания виртуального окружения можно воспользоваться утилитой venv которая работает из коробки. Про это есть статья на хабре. Если вы больше привыкли к virtualenv или ее обертке virtualenvwrapper то тут есть два варианта: либо поставить плагин к *pyenv*, или использовать их к той версии python с которой вы работаете. Соответственно если выбрать первый вариант, то созданные вами окружения будут добавлены к вашим версиям python и доступны через команду:

```
$ pyenv versions
```

Добавить плагин легко, просто клонируем его из репозитория pyenv-virtualenv или pyenv-virtualenvwrapper:

```
$ mkdir -p ~/.pyenv/plugins
$ git clone git://github.com/yyuu/pyenv-virtualenv.git
~/.pyenv/plugins/pyenv-virtualenv
```

```
$ git clone git://github.com/yyuu/pyenv-virtualenvwrapper.git
~/.pyenv/plugins/pyenv-virtualenvwrapper
```

Пример использования можно посмотреть в документации для pyenv-virtualenv и pyenv-virtualenvwrapper.
Все, а дальше пользуйтесь, как вам привычнее.

Пример использования

```
$ pyenv versions
* system
  2.7.5
  3.3.2

$ mkdir -p ~/test_project/prj_for_2.7.5 && cd
~/test_project/prj_for_2.7.5
$ pyenv virtualenv 2.7.5 my-virtualenv-2.7.5
$ pyenv local my-virtualenv-2.7.5
$ pip install django==1.4
$ pip freeze
Django==1.4
wsgiref==0.1.2
$ python --version
Python 2.7.5

$ mkdir -p ~/test_project/test_project && cd
~/test_project/test_project
$ pyenv virtualenv 3.3.2 my-virtualenv-3.3.2
$ pyenv local my-virtualenv-3.3.2
$ pip install django==1.5
$ pip freeze
Django==1.5
$ python --version
Python 3.3.2
```

Теперь находясь в директории проекта можно запускать скрипт от нужной версии python не прилагая никаких действий. *pyenv* создает в директории файл *.python-version* который содержит в себе информацию о том какую версию python с каким окружение

использовать для данного проекта.

Полезные ссылки

github.com/utahta/pythonbrew
github.com/yyuu/pyenv
github.com/yyuu/pyenv-installer
github.com/yyuu/pyenv-virtualenv
github.com/yyuu/pyenv-virtualenvwrapper
docs.python.org/dev/library/venv.html
www.virtualenv.org/en/latest
virtualenvwrapper.readthedocs.org/en/latest

Конвертация типов в Boost.Python. Делаем преобразование между привычными типами C++ и Python 4

Qualab • 10 февраля 2013 в 12:24

sjFilemanager — Reloaded. AJAX + PHP файловый менеджер версия 1.0 бета 50

serjoga • 29 октября 2011 в 21:33

Emacs и Python, Python и Emacs 56

eLLoco • 7 декабря 2008 в 22:05

Популярное за сутки

Яндекс открывает Алису для всех разработчиков. Платформа Яндекс.Диалоги (бета) 69

BarakAdama • вчера в 10:52

Почему следует игнорировать истории основателей успешных стартапов 20

ПЕРЕВОД

m1rko • вчера в 10:44

Как получить телефон (почти) любой красотки в Москве, или интересная особенность MT_FREE 24

ИЗ ПЕСОЧНИЦЫ

cab404 • вчера в 20:27

Java и Project Reactor 10

zealot_and_frenzy • вчера в 10:56

Пользовательские агрегатные и оконные функции в PostgreSQL и Oracle 6

erogov • вчера в 12:46

Лучшее на Geektimes

Как фермеры Дикого Запада организовали телефонную сеть на колючей проволоке 31

NAGru • вчера в 10:10

Энтузиаст сделал новую материнскую плату для ThinkPad X200s 49

alizar • вчера в 15:32

Кто-то посылает секс-игрушки с Amazon незнакомцам. Amazon не знает, как их остановить 85

Pochtoycom • вчера в 13:06

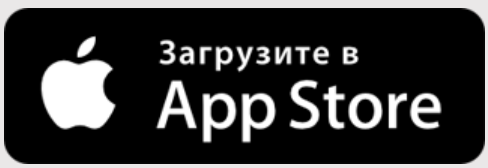
Илон Маск продолжает убеждать в необходимости создания колонии людей на Марсе 139

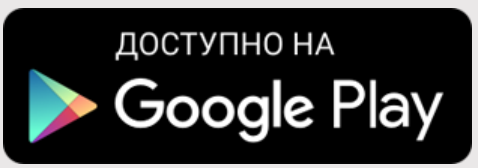
marks • вчера в 14:19

Дела шпионские (часть 1) 16

TashaFridrih • вчера в 13:16

Мобильное приложение





Полная версия