一、virtualenv 虚拟环境安装

virtualenv 工具可以用来在操作系统中创建一个虚拟的Python环境。这个环境是独立的、隔离的，拥有自己的环境安装目录（而不是把所有的第三方包等都安装在 /usr/lib/python3.4/site-packages 目录（Python3.4的默认Linux安装目录）下）。

它可以用来解决Python开发和运行过程中的依赖项以及相关版本问题，而不必和其他的Python环境以及全局的Python环境发生冲突。

virtualenv是pyvenv的第三方替代品（和前身）。它允许在3.4之前的Python版本（不提供pyvenv或者不能自动安装pip环境）上使用虚拟环境。

1、安装 virtualenv

建议使用pip工具安装virtualenv包：

```
$ pip install virtualenv
```

2、创建虚拟环境

安装完成后，可以使用 virtualenv 命令创建放置虚拟环境的目录：

```
$ virtualenv [OPTIONS] [虚拟环境名称]
```

```
[root@Sc2_jtgqspth_1 python_project]# virtualenv ENV
Using base prefix '/usr/local/python34'
New python executable in /root/python_project/ENV/bin/python3.4
Also creating executable in /root/python_project/ENV/bin/python
Installing setuptools, pip, wheel...done.
```

创建原理：

在目录 ENV 里会初始化虚拟环境的相关目录和文件，包括Python语言本身的环境以及 pip 等相关程序都会在这个目录里创建（拷贝）一份新的。

```
[root@Sc2_jtgqspth_1 python_project]# ll ENV/
total 16
drwxr-x--- 2 root root 4096 Oct 23 15:51 bin
drwxr-x--- 2 root root 4096 Oct 23 15:51 include
drwxr-x--- 3 root root 4096 Oct 23 15:51 lib
-rw-r----- 1 root root   60 Oct 23 15:51 pip-selfcheck.json
```

ENV/lib 和 ENV/include 目录中包含了虚拟环境ENV使用的库文件。在虚拟环境中安装的第三方包会安装在 ENV/lib/python3.4/site-packages 目录下。

ENV/bin 目录里面放置了可执行文件，在里面有新的Python 包中的可执行程序，包括pip等相关工具。

OPTIONS：

--no-site-packages 参数：

默认情况下，虚拟环境会依赖系统环境中的 sit package，就是说系统中已经安装好的第三方 package

也会安装在虚拟环境中，如果不想依赖这些 package，那么可以加上 --no-site-packages 参数建立虚拟环境。

$ virtualenv --no-site-packages [虚拟环境名称]

```
[root@Sc2_jtgqspth_1 test]# virtualenv --no-site-packages ENV
Using base prefix '/usr/local/python34'
New python executable in /root/python_project/test/ENV/bin/python3.4
Also creating executable in /root/python_project/test/ENV/bin/python
Installing setuptools, pip, wheel...done.
```

--system-site-packages 参数：

如果你使用 --system-site-packages 参数创建虚拟环境，你创建的虚拟环境会继承 /usr/local/python34/lib/python3.4/site-packages 。

$ virtualenv --system-site-packages ENV

```
[root@Sc2_jtgqspth_1 python_project]# virtualenv --system-site-packages ENV
Using base prefix '/usr/local/python34'
New python executable in /root/python_project/ENV/bin/python3.4
Also creating executable in /root/python_project/ENV/bin/python
Installing setuptools, pip, wheel...done.
```

--extra-search-dir 参数：

如果你使用 --extra-search-dir 参数创建虚拟环境，这允许你提供自己的setuptools版本，而不是使用虚拟环境中嵌入的版本。

$ virtualenv --extra-search-dir=/path/to/distributions ENV

/path/to/distributions 参数：

指定一个包含setuptools以及pip、wheels等的目录。virtualenv命令会在指定的目录中，寻找wheels，但是会使用pip的标准的算法来选择安装的wheel。

除了指定的目录，搜索顺序还包含：

1、相对于virtualenv.py的 virtualenv_support 目录；

2、virtualenv.py所在的目录；

3、当前目录；

--version 显示软件的版本号：

virtualenv --version

```
[root@Sc2_jtgqspth_1 ~]# virtualenv --version
15.1.0
```

-h, --help 显示帮助信息：

virtualenv -h

```
[root@Sc2_jtgqspth_1 ~]# virtualenv -h
Usage: virtualenv [OPTIONS] DEST_DIR

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
```

-p PYTHON_EXE, --python=PYTHON_EXE 指定生成的虚拟环境使用的Python解释器（默认的使用的是安装virtualenv的Python解释器）：

virtualenv -p /usr/bin/python2.6 ENV

```
[root@Sc2_jtgqspth_1 ~]# virtualenv -p /usr/bin/python2.6 ENV
Running virtualenv with interpreter /usr/bin/python2.6
New python executable in /root/ENV/bin/python2.6
Also creating executable in /root/ENV/bin/python
Installing setuptools, pip, wheel...done.
[root@Sc2_jtgqspth_1 ~]# ll ENV/bin/p
pip             pip2.6          python2         python-config
pip2            python          python2.6
```

--no-setuptools 在新建的虚拟环境中不安装工具包

virtualenv --no-setuptools ENV

```
[root@Sc2_jtgqspth_1 ~]# virtualenv --no-setuptools ENV
Using base prefix '/usr/local/python34'
New python executable in /root/ENV/bin/python3.4
Also creating executable in /root/ENV/bin/python
Installing pip, wheel...done.
[root@Sc2_jtgqspth_1 ~]# ll ENV/bin/
total 8644
-rw-r----- 1 root root    2068 Oct 24 09:41 activate
-rw-r----- 1 root root    1010 Oct 24 09:41 activate.csh
-rw-r----- 1 root root    2164 Oct 24 09:41 activate.fish
-rw-r----- 1 root root    1137 Oct 24 09:41 activate_this.py
-rwxr-xr-x 1 root root     213 Oct 24 09:41 pip
-rwxr-xr-x 1 root root     213 Oct 24 09:41 pip3
-rwxr-xr-x 1 root root     213 Oct 24 09:41 pip3.4
lrwxrwxrwx 1 root root       9 Oct 24 09:41 python -> python3.4
lrwxrwxrwx 1 root root       9 Oct 24 09:41 python3 -> python3.4
-rwxr-xr-x 1 root root 8811815 Oct 24 09:41 python3.4
-rwxr-xr-x 1 root root    2327 Oct 24 09:41 python-config
-rwxr-xr-x 1 root root     220 Oct 24 09:41 wheel
```

--no-pip 在新建的虚拟环境中不安装pip：

virtualenv --no-pip ENV

```
[root@Sc2_jtgqspth_1 ~]# virtualenv --no-pip ENV
Using base prefix '/usr/local/python34'
New python executable in /root/ENV/bin/python3.4
Also creating executable in /root/ENV/bin/python
Installing setuptools, wheel...done.
[root@Sc2_jtgqspth_1 ~]# ls ENV/bin/
activate      activate.fish     easy_install      python    python3.4      wheel
activate.csh  activate_this.py  easy_install-3.4  python3   python-config
```

--no-wheel 在新建的虚拟环境中不安装wheel：

virtualenv --no-wheel ENV

```
[root@Sc2_jtgqspth_1 ~]# virtualenv --no-wheel ENV
Using base prefix '/usr/local/python34'
New python executable in /root/ENV/bin/python3.4
Also creating executable in /root/ENV/bin/python
Installing setuptools, pip...done.
[root@Sc2_jtgqspth_1 ~]# ls ENV/bin/
activate      activate.fish     easy_install      pip   pip3.4  python3    python-config
activate.csh  activate_this.py  easy_install-3.4  pip3  python  python3.4
```

--extra-search-dir=DIR 指定安装新的虚拟环境时寻找工具包、pip等的目录路径，这个参数可以用来多次指定：

virtualenv --extra-search-dir=/usr/local/python34/bin/ ENV

```
[root@Sc2_jtgqspth_1 bin]# virtualenv --extra-search-dir=/usr/local/python34/bin/ ENV
Using base prefix '/usr/local/python34'
New python executable in /usr/bin/ENV/bin/python3.4
Also creating executable in /usr/bin/ENV/bin/python
Installing setuptools, pip, wheel...done.
[root@Sc2_jtgqspth_1 bin]# ll ENV/bin/
total 8652
-rw-r----- 1 root root    2071 Oct 24 09:49 activate
-rw-r----- 1 root root    1013 Oct 24 09:49 activate.csh
-rw-r----- 1 root root    2167 Oct 24 09:49 activate.fish
-rw-r----- 1 root root    1137 Oct 24 09:49 activate_this.py
-rwxr-xr-x 1 root root     244 Oct 24 09:49 easy_install
-rwxr-xr-x 1 root root     244 Oct 24 09:49 easy_install-3.4
-rwxr-xr-x 1 root root     216 Oct 24 09:49 pip
-rwxr-xr-x 1 root root     216 Oct 24 09:49 pip3
-rwxr-xr-x 1 root root     216 Oct 24 09:49 pip3.4
lrwxrwxrwx 1 root root       9 Oct 24 09:49 python -> python3.4
lrwxrwxrwx 1 root root       9 Oct 24 09:49 python3 -> python3.4
-rwxr-xr-x 1 root root 8811815 Oct 24 09:49 python3.4
-rwxr-xr-x 1 root root    2330 Oct 24 09:49 python-config
-rwxr-xr-x 1 root root     223 Oct 24 09:49 wheel
```

--prompt=PROMPT 可以为创建的虚拟环境指定提示前缀：

virtualenv --prompt="测试" test

```
[root@Sc2_jtgqspth_1 python_project]# virtualenv --prompt="测试" test
Using base prefix '/usr/local/python34'
New python executable in /root/python_project/test/bin/python3.4
Also creating executable in /root/python_project/test/bin/python
Installing setuptools, pip, wheel...done.
```

```
[root@Sc2_jtgqspth_1 python_project]# source test/bin/activate
测试[root@Sc2_jtgqspth_1 python_project]#
```

3、启动环境

在新创建的虚拟环境目录中，有一个启动脚本 ENV/bin/activate ，直接使用 source 命令来启动我们新创建的虚拟环境。

$ source ENV/bin/activate

```
[root@Sc2_jtgqspth_1 python_project]# source ENV/bin/activate
(ENV) [root@Sc2_jtgqspth_1 python_project]#
```

这条命令修改了当前用户的环境变量 $PATH，将我们新建的虚拟环境目录 ENV/bin/作为第一个元素添加到了 $PATH 中。因为在Linux系统中输入命令时，命令可执行文件的搜索路径是根据 $PATH 中配置的路径的先后顺序来的，所以接下来你执行的所有命令都将是 ENV/bin/ 中的命令（如果此目录中存在这个命令的话），新安装的模块都只会安装到该目录（ENV）中去。

```
[root@Sc2_jtgqspth_1 python_project]# echo $PATH
/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin
```

```
(ENV) [root@Sc2_jtgqspth_1 python_project]# echo $PATH
/root/python_project/ENV/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/ro
ot/bin
```

active脚本也会修改当前用户的命令行显示，在头部会显示出当前使用的虚拟环境。

4、退出和删除环境

如果要退出当前虚拟环境，使用 deactivate 命令：

$ deactivate

```
(ENV) [root@Sc2_jtgqspth_1 bin]# deactivate
[root@Sc2_jtgqspth_1 bin]#
```

如果需要删除我们创建的虚拟环境，只需要退出，并删除创建的文件夹即可：

$ deactivate

$ rm -r /path/to/ENV

```
(ENV) [root@Sc2_jtgqspth_1 python_project]# deactivate
[root@Sc2_jtgqspth_1 python_project]# rm -r ENV/
```

二、安装 virtualenvwrapper 扩展包

Virtaulenvwrapper是virtualenv的扩展包，可以把新创建的环境记录下来，不需要每次启动虚拟环境时都执行一遍source命令，可以更方便的管理虚拟环境。

它可以实现：

1、将所有虚拟环境整合在一个目录下

2、管理（新增，删除，复制）虚拟环境

3、快速切换虚拟环境

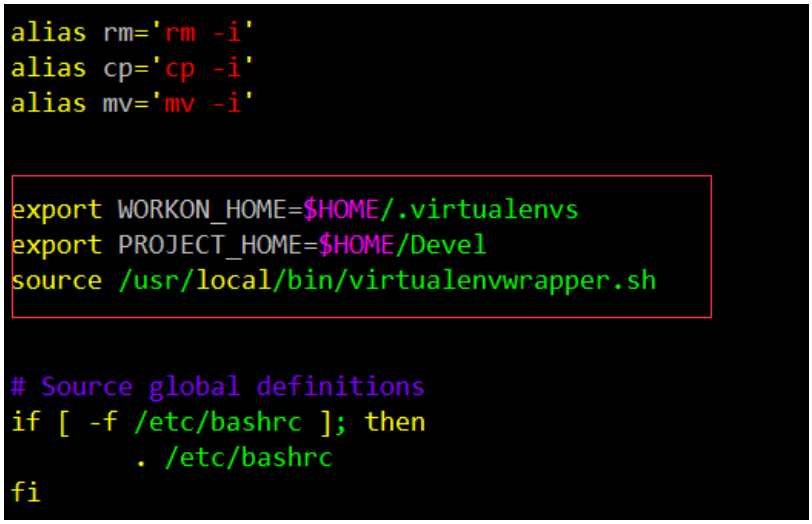
1、安装 virtualenvwrapper ：

1.1 建议适用 pip 安装：

```
$ pip install virtualenvwrapper
```

1.2 进入 用户主目录，打开 .bashrc 文件，添加如下代码：

```
export WORKON_HOME=$HOME/.virtualenvs
export PROJECT_HOME=$HOME/Devel
source /usr/local/bin/virtualenvwrapper.sh
```



WORKON_HOME：

告诉 virtualenvwrapper 在哪里放置你的虚拟环境，默认是在 $HOME/.virtualenvs 目录下。如果这个目录不存在，virtualenvwrapper运行的时候会自动创建它。

PROJECT_HOME ：

告诉virtualenvwrapper在哪里存放你的项目的工作目录。

1.3 然后执行 source 命令，使刚添加的代码生效：

```
$ source .bashrc
```

2、使用 virtualenvwrapper

虚拟环境相关操作

workon:列出虚拟环境列表

mkvirtualenv [虚拟环境名称]:新建虚拟环境

workon [虚拟环境名称]:切换虚拟环境

rmvirtualenv [虚拟环境名称]:删除虚拟环境

deactivate: 离开虚拟环境

2.1 使用 workon 命令，列出可以使用的虚拟环境：

$ workon

```
[root@Sc2_jtgqspth_1 ~]# workon
cms
```

2.2 使用 mkvirtualenv 命令，创建新的虚拟环境：

mkvirtualenv [-a project_path] [-i package] [-r requirements_file] [virtualenv options] ENVNAME

除了-a ， -i，-r，-h 所有的命令中的选项都会直接传递给virtualenv，新的虚拟环境在初始化后会自动激活（也就是进入该虚拟环境）。

$ mkvirtualenv ENV

```
[root@Sc2_jtgqspth_1 ~]# mkvirtualenv ENV
Using base prefix '/usr/local/python34'
New python executable in /root/.virtualenvs/ENV/bin/python3.4
Also creating executable in /root/.virtualenvs/ENV/bin/python
Installing setuptools, pip, wheel...done.
virtualenvwrapper.user_scripts creating /root/.virtualenvs/ENV/bin/predeactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/ENV/bin/postdeactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/ENV/bin/preactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/ENV/bin/postactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/ENV/bin/get_env_details
```

-a 选项：可以将一个现有的项目关联到新建的虚拟环境

$ mkvirtualenv -a phone/ phone

```
(ENV) [root@Sc2_jtgqspth_1 test]# mkvirtualenv -a phone/ phone
Using base prefix '/usr/local/python34'
New python executable in /root/.virtualenvs/phone/bin/python3.4
Also creating executable in /root/.virtualenvs/phone/bin/python
Installing setuptools, pip, wheel...done.
virtualenvwrapper.user_scripts creating /root/.virtualenvs/phone/bin/predeactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/phone/bin/postdeactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/phone/bin/preactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/phone/bin/postactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/phone/bin/get_env_details
Setting project for phone to /root/python_project/test/phone
(phone) [root@Sc2_jtgqspth_1 phone]#
```

-i 选项：可以在虚拟环境创建后安装一个或多个包

-r选项：可以指定一个列出了要安装那些包的文件，这个参数相当于pip 的 -r 参数。

2.3 使用 mktmpenv 命令自动生成一个唯一的名称：

mktmpenv [(-c|--cd)|(-n|--no-cd)] [VIRTUALENV_OPTIONS]

$ mktmpenv

```
[root@Sc2_jtgqspth_1 ~]# mktmpenv
Using base prefix '/usr/local/python34'
New python executable in /root/.virtualenvs/tmp-a8f6c8fa58d2be8/bin/python3.4
Also creating executable in /root/.virtualenvs/tmp-a8f6c8fa58d2be8/bin/python
Installing setuptools, pip, wheel...done.
virtualenvwrapper.user_scripts creating /root/.virtualenvs/tmp-a8f6c8fa58d2be8/bin/predeactiv
ate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/tmp-a8f6c8fa58d2be8/bin/postdeacti
vate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/tmp-a8f6c8fa58d2be8/bin/preactivat
e
virtualenvwrapper.user_scripts creating /root/.virtualenvs/tmp-a8f6c8fa58d2be8/bin/postactiva
te
virtualenvwrapper.user_scripts creating /root/.virtualenvs/tmp-a8f6c8fa58d2be8/bin/get_env_de
tails
This is a temporary environment. It will be deleted when you run 'deactivate'.
(tmp-a8f6c8fa58d2be8) [root@Sc2_jtgqspth_1 tmp-a8f6c8fa58d2be8]# ll
```

2.4 lsvirtualenv 命令详细的列出我们创建的虚拟环境

$ lsvirtualenv

```
[root@Sc2_jtgqspth_1 ~]# lsvirtualenv
cms
===


ENV
===


phone
=====
```

2.5 showvirtualenv 显示某个虚拟环境的细节描述

showvirtualenv [env]

$ showvirtualenv cms

```
[root@Sc2_jtgqspth_1 ~]# showvirtualenv cms
```

2.6 rmvirtualenv 删除 WORKON_HOME 里的一个虚拟环境

在你删除虚拟环境之前，必须先离开这个虚拟环境。

rmvirtualenv ENVNAME

$ rmvirtualenv test

```
[root@Sc2_jtgqspth_1 ~]# workon
cms
ENV
phone
test
[root@Sc2_jtgqspth_1 ~]# rmvirtualenv test
Removing test...
[root@Sc2_jtgqspth_1 ~]# rmvirtualenv phone
Removing phone...
[root@Sc2_jtgqspth_1 ~]# workon
cms
ENV
```

2.7 allvirtualenv 在WORKON_HOME里的所有虚拟环境中执行一条命令：

```
allvirtualenv command with arguments
$ allvirtualenv pip install -U pip
```

2.8 workon 切换一个虚拟环境

```
workon environment_name
$ workon cms
```

```
[root@Sc2_jtgqspth_1 ~]# workon cms
(cms) [root@Sc2_jtgqspth_1 ~]#
```

如果后面没有给出虚拟环境的名称，则会列出可以使用的虚拟环境

```
[root@Sc2_jtgqspth_1 ~]# workon
cms
```

2.9 deactivate 退出当前使用的虚拟环境

```
(cms) [root@Sc2_jtgqspth_1 ~]# deactivate
[root@Sc2_jtgqspth_1 ~]#
```

2.10 virtualenvwrapper 打印出可以使用的命令，及简单的描述信息

```
[root@Sc2_jtgqspth_1 ~]# virtualenvwrapper

virtualenvwrapper is a set of extensions to Ian Bicking's virtualenv
tool.  The extensions include wrappers for creating and deleting
virtual environments and otherwise managing your development workflow,
making it easier to work on more than one project at a time without
introducing conflicts in their dependencies.

For more information please refer to the documentation:

    http://virtualenvwrapper.readthedocs.org/en/latest/command_ref.html

Commands available:

  add2virtualenv: add directory to the import path

  allvirtualenv: run a command in all virtualenvs

  cdproject: change directory to the active project

  cdsitepackages: change to the site-packages directory

  cdvirtualenv: change to the $VIRTUAL_ENV directory

  cpvirtualenv: duplicate the named virtualenv to make a new one

  lssitepackages: list contents of the site-packages directory

  lsvirtualenv: list virtualenvs

  mkproject: create a new project directory and its associated virtualenv

  mktmpenv: create a temporary virtualenv

  mkvirtualenv: Create a new virtualenv in $WORKON_HOME
```