今天硬着头皮做完了python自动测试的习题，发现还真是好用，相当于顺着自己的思路写程序(测试部分代码写的好的话)，所以把这部分整理一下。

从源头开始，配置虚拟环境和创建骨架，因为要使用nosetest实现测试还是需要一些结构上的条件的，以下都是基于windows 10的操作：

配置虚拟环境

在win10上配置虚拟环境首先要确定只安装了一个版本的python,在power shell里面输入:

```
cd ~
python
```

如果输出的只有一个python版本信息就可以了，否则要卸载一个。
然后确认基本的安装：

```
pip list
```

至少要有`pip`和`setuptools`就可以了，有多的也无所谓。

然后要安装virtualenv来设置简单的虚拟环境：

```
pip install virtualenv
```

安装好了之后需要创建一个`.venvs`文件夹，里面装上虚拟环境：

```
mkdir .venvs
virtualenv --system-site-packages .venvs/lpthw
```

这样就创建了一个.venvs文件夹，用来储存不同的虚拟环境，第二句创建了一个虚拟环境，叫做`lpthw`。
在虚拟环境里面可以任意安装软件包而对主系统没有影响。
之后要激活虚拟环境：

```
..venvs\lpthw\Scripts\activate
```

这样就把当前的shell设为使用lpthw虚拟环境，要想使用虚拟环境，就要每次都先执行上面那条命令，成功后命令提示符前面会出现一个(lpthw)。

建立骨架

首先用以下命令创建骨架目录的结构：

```
mkdir projects
cd projects/
mkdir skeleton
cd skeleton
mkdir bin NAME tests docs
```

其中，projects是储存项目的文件夹，skeleton是新项目的基础目录，NAME是项目的主模块，创建新项目的时候改成项目名称。
然后设置一些初始文件：

```
new-item -type file NAME/__init__.py
new-item -type file tests/__init__.py
```

然后建立一个setup.py文件，安装项目的时候会用到：

```
try:
    from setuptools import setup
except ImportError:
    from distutils.core import setup

config = {
    'description': 'My project',
    'author': 'Chunar',
    'url': 'URL to get it at.',
    'download_url': 'Where to download it.',
    'author_email': 'My email.',
    'version': '0.1',
    'install_requires': ['nose'],
    'scripts': [],
    'name': 'projectname'
}

setup(**config)
```

把里面的信息换成自己的（只做测试的话没必要）。

然后需要建一个测试专用的骨架文件叫做tests/NAME_tests.py:

```
from nose.tools import *
import NAME_tests

def setup():
  print("SETUP!")

def teardown():
  print("TEAR DOWN!")

def test_basic():
  print("I RAN!")
```

这时你的skeleton文件夹应该是这样的：

```
skeleton/
  NAME/
    __init__.py
  bin/
  docs/
  setup.py
  tests/
    NAME_tests.py
    __init__.py
```

这样以后需要建立新项目的时候就简单多了：

1.复制skeleton文件夹，将名字改成新项目的名字；
2.将NAME目录改为项目的名字；
3.编辑 `setup.py` ，输入新项目的相关信息；
4.重命名 `tests/NAME_tests.py` ，把NAME换成模块的名字；
5.使用nosetests检查有无错误；
6.开始写代码。

关于自动测试

为了实现测试需要安装nose软件包：

> pip install nose

然后就可以在上面的skeleton目录下注意是**skeleton目录下**，在命令行中输入 `nosetests` 进行测试

测试成功的效果大概是这样的：

```
$ nosetests

----------------------------
Ran 1 tests in 0.001s

OK
```

复制skeleton到新项目并准备测试之前，还需要做一些事情：

1.首先当然是复制骨架到新项目；
2.将所有 `NAME` 的部分都改为项目名称；
3.将文件内部的所有 `NAME` 都改为项目名称（或者自己起的新名字）；
4.删除所有*.pyc文件。

自己的一点小经验：

写测试文件的思路：主要是使用 `assert_equal()` ，里面两个部分，一个是跟着程序走，程序运行得出的结果；第二部分是自己设计的应有的结果。 `assert_equal()` 就是测试是否相等的函数。

——by 秦小炅 2018.9.9