A local /bin Directory

- Adding a local directory for executable files.
- Adding local directories for python modules

Hamilton Python Users Group
11 March 2019
Ian Stewart

$PATH - /usr/local/bin/

```
$ echo $PATH
Bash:
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:
/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:
$
```

Write a python program

```
$ touch my_python_prog

$ nano my_python_prog

$ cat my_python_prog
#!/usr/bin/env python3
#!
print("my_python_prog prints out this message")

$ ls -l my_python_prog
-rw-rw-r-- 1 ian ian 75 Mar 10 17:09
my_python_prog
```

Run Program

```
$ python3 my_python_prog
my_python_prog prints out this message

$ my_python_prog
my_python_prog: command not found

$ ./my_python_prog
bash: ./my_python_prog: Permission denied

$ sudo ./my_python_prog
[sudo] password for ian:
sudo: ./my_python_prog: command not found
```

Use chmod +x to run program. But...

```
$ chmod +x my_python_prog

$ ls -l my_python_prog
-rwxrwxr-x 1 ian ian 75 Mar 10 17:09 my_python_prog

$ ./my_python_prog
my_python_prog prints out this message

$ mkdir another_folder

$ cd another_folder

$ ./my_python_prog
bash: ./my_python_prog: No such file or directory
```

Need priv to copy to /usr/local/bin/

```
$ ls -l /usr/local/
drwxr-xr-x 2 root root 4096 May 15  2018 bin
...snip...

$ ls -l my_python_prog
-rwxrwxr-x 1 ian ian 75 Mar 10 17:09 my_python_prog

$ cp my_python_prog /usr/local/bin
cp: cannot create regular file
'/usr/local/bin/my_python_prog': Permission denied

$ sudo cp my_python_prog /usr/local/bin
[sudo] password for ian:
$
```

Try out program in/usr/local/bin

```
$ ls -l /usr/local/bin/my_python_prog
-rwxr-xr-x 1 root root 75 Mar 10 19:40
/usr/local/bin/my_python_prog

$ ls -l my_python_prog
-rwxrwxr-x 1 ian ian 75 Mar 10 17:09 my_python_prog

$ mv my_python_prog my_python_prog_local
$ ls -l my_python_prog_local
-rwxrwxr-x 1 ian ian 75 Mar 10 17:09
my_python_prog_local

$ my_python_prog
my_python_prog prints out this message
```

Program runs OK. Unable to modify program...

```
$ my_python_prog
my_python_prog prints out this message
```

```
$ nano /usr/local/bin/my_python_prog
```

```
  GNU nano 2.9.3                                  /usr/local/bin/my_python_prog

#!/usr/bin/env python3
#!
print("my_python_prog prints out this message")
print("changed the program by editing it with nano")
```

```
File Name to Write: /usr/local/bin/my_python_prog
^G Get Help                M-D DOS Format                M-A
^C Cancel                  M-M Mac Format                M-P
```

```
[ Error writing /usr/local/bin/my_python_prog: Permission denied ]
```

Need priv to modify my program in /usr/local/bin/

```
$ my_python_prog
my_python_prog prints out this message
```

```
$ sudo nano /usr/local/bin/my_python_prog
```

```
  GNU nano 2.9.3                                  /usr/local/bin/my_python_prog

#!/usr/bin/env python3
#!
print("my_python_prog prints out this message")
print("Can I change the program this time")
```

```
File Name to Write: /usr/local/bin/my_python_prog
```

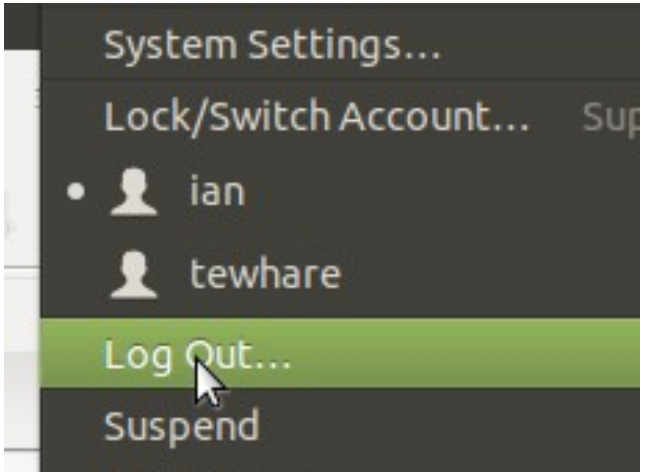
```
$ my_python_prog
my_python_prog prints out this message
Can I change the program this time
```

Examine the accounts .profile file...

```
$ cd ~/

$ cat .profile
# ~/.profile: executed by the command interpreter
# for login shells.
# This file is not read by bash(1), if
# ~/.bash_profile or ~/.bash_login exists.
... snip...
# set PATH so it includes user's private bin if it
# exists
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi
```

Make a bin folder. Move program to bin folder. Logout of account...

```
$ mkdir bin

$ mv my_python_prog_local bin/my_python_prog_local

$ ls -l bin/my_python_prog
-rwxrwxr-x 1 ian ian 75 Mar 10 20:33
bin/my_python_prog_local
```



Log back in to account. Run / Modify program...

```
$ echo $PATH
bash: /home/ian/bin:/usr/local/sbin:/usr/local/bin:...

$ my_python_prog_local
my_python_prog prints out this message

$ cd bin
$ nano my_python_prog_local
```

```
  GNU nano 2.9.3                                   my_python_prog_local

#!/usr/bin/env python3
#!
print("my_python_prog prints out this message")
print("Changed my_python_prog_local with normal priv")

File Name to Write: my_python_prog_local
```

Get out of bin folder. Can run modified program...

```
$ cd ~/

$ my_python_prog_local
my_python_prog prints out this message
Changed my_python_prog_local with normal priv
```

Create bin-x sub-directories off bin...

```
$ cd ~/
$ cd bin
$ ls -l
total 4
-rwxrwxr-x 1 ian ian 129 Mar 10 20:46
my_python_prog_local

$ mkdir bin-python
$ mkdir bin-bash
$ mkdir bin-development

$ ls
bin-bash  bin-development  bin-python
my_python_prog_local
```

Move program to sub-directory. Fails...

```
$ my_python_prog_local
my_python_prog prints out this message
Changed my_python_prog_local with normal priv

$ mv my_python_prog_local bin-python/
$ ls -l bin-python
total 4
-rwxrwxr-x 1 ian ian 129 Mar 10 20:46
my_python_prog_local

$ my_python_prog_local
bash: /home/ian/bin/my_python_prog_local: No such
file or directory
```

Add code to .profile. Then Logout and Login...

```
if [ -d "$HOME/bin" ]
then
    for file in ${HOME}/bin/bin-*
    do
        if [ -d "$file" ]
        then
             PATH="$file:$PATH"
        fi
    done
fi
```

$PATH after logout / login...

```
$ echo $PATH
bash:
/home/ian/bin:/usr/local/sbin:/usr/local/bin:/...

$# Equivalent to Logout / Login
$ source ~/.profile

$ echo $PATH
bash: /home/ian/bin/bin-python:/home/ian/bin/bin-
development:/home/ian/bin/bin-bash:/home/ian/bin:/
usr/local/sbin:/usr/local/bin:/...
```

Does program run from bin/bin-python/

```
$ cd ~/

$ ls bin/bin-python/
my_python_prog_local

$ my_python_prog_local
my_python_prog prints out this message
Changed my_python_prog_local with normal priv
```

pythagoras_1 program in bin/bin-python/

```
1 #!/usr/bin/env python3
2 #!
3 # pythagoras_1
4 import math
5 adjacent = input("Enter the length of the adjacent side: ")
6 opposite = input("Enter the length of the opposite side: ")
7
8 hypotenuse = math.sqrt(float(adjacent)**2 + float(opposite)**2)
9 print("Length of hypotenuse: {:g}". format(hypotenuse))
```

chmod + x to run from bash prompt...

```
$ pythagoras_1
bash: /home/ian/bin/bin-python/pythagoras_1: Permissio
denied

$ ls -l bin/bin-python/
total 8
-rwxrwxr-x 1 ian ian 129 Mar 10 20:46 my_python_prog_l
-rw-rw-r-- 1 ian ian 294 Mar 10 22:45 pythagoras_1

$ chmod +x bin/bin-python/pythagoras_1
$ ls -l bin/bin-python/
total 8
-rwxrwxr-x 1 ian ian 129 Mar 10 20:46 my_python_prog_l
-rwxrwxr-x 1 ian ian 294 Mar 10 22:45 pythagoras_1
```

Program now runs from $ prompt

```
$ cd ~/

$ pythagoras_1
Enter the length of the adjacent side: 5
Enter the length of the opposite side: 12
Length of hypotenuse: 13
```

Summary, so far...

- Python, bash, etc. scripts can be run as bash commands.
- Scripts are normally in folder /usr/local/bin/
- Scripts need priv to copy there and to edit.
- Many computers have only one user.
- Can create a /home/USER/bin/ folder.
- .profile adds ~/bin to $PATH
- Scripts in ~/bin upon having chmod +x can be executed from any folder in the User's account.
- Scripts in sub-directories of ~/bin will not execute.
- Modify .profile to support some sub-directories of ~/bin running python or bash scripts.

Continue with...

- Sub-folders as executable.
- Python modules in sub-directories.
- $PYTHONPATH to local directory.

Pythagorean theorem using a function...

```python
#!/usr/bin/env python3
#!
# pythagoras_2
import math

def calculate_hypotenuse(adjacent=3, opposite=4):
    # Use Pythagorean theorem to calculate hypotenuse.
    hypotenuse = math.sqrt(float(adjacent)**2 + float(opposite)**2)
    return hypotenuse

if __name__ == "__main__":
    adjacent = input("Enter the length of the adjacent side: ")
    opposite = input("Enter the length of the opposite side: ")

    hypotenuse = calculate_hypotenuse(adjacent, opposite)
    print("Length of hypotenuse: {:g}". format(hypotenuse))
```

Run pythagoras_2

```
$ python3 bin/bin-python/pythagoras_2
Enter the length of the adjacent side: 5
Enter the length of the opposite side: 12
Length of hypotenuse: 13
```

Convert program with function to a module

```python
#!/usr/bin/env python3
#!
# pythagoras_theorem.py
import math

def calculate_hypotenuse(adjacent=3, opposite=4):
    # Use Pythagorean theorem to calculate hypotenuse.
    hypotenuse = math.sqrt(float(adjacent)**2 + float(opposite)**2)
    return hypotenuse

if __name__ == "__main__":
    # For testing...
    #adjacent = input("Enter the length of the adjacent side: ")
    #opposite = input("Enter the length of the opposite side: ")
    #hypotenuse = calculate_hypotenuse(adjacent, opposite)
    hypotenuse = calculate_hypotenuse()
    print("Length of hypotenuse: {:g}". format(hypotenuse))
```

pythagoras_theorem.py module can be run on its own to test it works OK...

```
$ python3 bin/bin-python/pythagoras_theorem.py
Length of hypotenuse: 5
```

Create pythagoras_3 that calls pythagoras_theorem.py module

```python
#!/usr/bin/env python3
#!
# pythagoras_3
import pythagoras_theorem

def main():
    adjacent = input("Enter the length of the adjacent side: ")
    opposite = input("Enter the length of the opposite side: ")
    hypotenuse = pythagoras_theorem.calculate_hypotenuse(adjacent, opposite)
    print("Length of hypotenuse: {:g}". format(hypotenuse))

if __name__ == "__main__":

    main()
```

Both pythagorus_3 and pythagorus_theorem.py files are in ~/bin/bin-python.

Run pythagoras_3

```
$ ls bin/bin-python/pythagoras_3 -l
-rw-rw-r-- 1 ian ian 386 Mar 10 23:24 pythagoras_3

$ chmod +x bin/bin-python/pythagoras_3

$ ls bin/bin-python/ -l
-rwxrwxr-x 1 ian ian  386 Mar 10 23:24 pythagoras_3
-rw-rw-r-- 1 ian ian  599 Mar 10 23:36
pythagoras_theorem.py

$ pythagoras_3
Enter the length of the adjacent side: 20
Enter the length of the opposite side: 21
Length of hypotenuse: 29
```

Does not require execute status

Run pythagoras_4 no if __name__=='__main__':

```python
#!/usr/bin/env python3
#!
# pythagoras_4
import pythagoras_theorem

adjacent = input("Enter the length of the adjacent side: ")
opposite = input("Enter the length of the opposite side: ")
hypotenuse = pythagoras_theorem.calculate_hypotenuse(adjacent, opposite)
print("Length of hypotenuse: {:g}". format(hypotenuse))
```

```
$ pythagoras_4
Enter the length of the adjacent side: 11
Enter the length of the opposite side: 60
Length of hypotenuse: 61
```

```python
#!/usr/bin/env python3
#!
# pythagoras_5. Pass arguments from bash
import pythagoras_theorem
import sys
if len(sys.argv) == 1:
    adjacent = input("Enter the length of the adjacent side: ")
    opposite = input("Enter the length of the opposite side: ")

if len(sys.argv) == 2:
    adjacent = sys.argv[1]
    opposite = input("Enter the length of the opposite side: ")

if len(sys.argv) == 3:
    adjacent = sys.argv[1]
    opposite = sys.argv[2]

hypotenuse = pythagoras_theorem.calculate_hypotenuse(adjacent, opposite)
print("Length of hypotenuse: {:g}". format(hypotenuse))
```

pythagoras_5
Bash passes arguments

pythagoras_5 – passing arguments

```
$ pythagoras_5
Enter the length of the adjacent side: 9
Enter the length of the opposite side: 40
Length of hypotenuse: 41

$ pythagoras_5 9
Enter the length of the opposite side: 40
Length of hypotenuse: 41

$ pythagoras_5 9 40
Length of hypotenuse: 41
```

Create a python library off bin-python

```
$ cd bin
$ cd bin-python
$ ls pythagoras_theorem.py -l
-rw-rw-r-- 1 ian ian 599 Mar 10 23:36
pythagoras_theorem.py

$ mkdir pylib

$ mv pythagoras_theorem.py pylib/

$ ls -l pylib/
total 4
-rw-rw-r-- 1 ian ian 599 Mar 10 23:36
pythagoras_theorem.py
```

pythagoras_6
import from pylib

```python
#!/usr/bin/env python3
#!
# pythagoras_6. Import from pylib
from pylib.pythagoras_theorem import calculate_hypotenuse

import sys
if len(sys.argv) == 1:
    adjacent = input("Enter the length of the adjacent side: ")
    opposite = input("Enter the length of the opposite side: ")

if len(sys.argv) == 2:
    adjacent = sys.argv[1]
    opposite = input("Enter the length of the opposite side: ")

if len(sys.argv) == 3:
    adjacent = sys.argv[1]
    opposite = sys.argv[2]

hypotenuse = calculate_hypotenuse(adjacent, opposite)
print("Length of hypotenuse: {:g}". format(hypotenuse))
```

pythagoras_6

```
$ pythagoras_6 6 8
Length of hypotenuse: 10

$ tree bin
bin
├── bin-bash
├── bin-development
└── bin-python
    ├── pythagoras_6
    ├── pylib
    │   ├── __pycache__
    │   │   └── pythagoras_theorem.cpython-36.pyc
    │   └── pythagoras_theorem.py
    │
```

Create a separate local python library directory

```
$ ~/
$ mkdir lib
$ cd lib
$ mkdir python

$ cd ~/
$ nano .profile
...
# Allow python modules to reside in local library.
export PYTHONPATH="$HOME/lib/python:$PYTHONPATH"
```

Logout and log back in.

pythagoras_7 – local library

```python
#!/usr/bin/env python3
#!
# pythagoras_7. Import from ~/lib/python/ in $PYTHONPATH
from pythagoras_theorem import calculate_hypotenuse

import sys
if len(sys.argv) == 1:
    adjacent = input("Enter the length of the adjacent side: ")
    opposite = input("Enter the length of the opposite side: ")

if len(sys.argv) == 2:
    adjacent = sys.argv[1]
    opposite = input("Enter the length of the opposite side: ")

if len(sys.argv) == 3:
    adjacent = sys.argv[1]
    opposite = sys.argv[2]

hypotenuse = calculate_hypotenuse(adjacent, opposite)
print("Length of hypotenuse: {:g}". format(hypotenuse))
```

Pythagoras_7 with updated $PYTHONPATH

```
$ pythagoras_7 6 8
Length of hypotenuse: 10

$ tree bin
/home/ian/
├── bin
│   ├── bin-bash
│   ├── bin-development
│   └── bin-python
│       └── pythagorus_7
├── lib
│   └── python
│       └── pythagoras_theorem.py
```

Comparison of imports...

pythagoras_6 with module in ~/bin/bin-python/pylib/

```
from pylib.pythagoras_theorem import calculate_hypotenus
```

pythagoras_7 with module in ~/lib/python/

```
from pythagoras_theorem import calculate_hypotenuse
```

Call is the same...

```
hypotenuse = calculate_hypotenuse(adjacent, opposite)
```

__end__