**Executing Shell Commands With subprocess.run**

Do pracy z procesami zewnętrznymi będziemy eksperymentować z modułem podprocesu z REPL. Główną funkcją, z którą będziemy pracować, jest funkcja subprocess.run, która zapewnia nam dużą elastyczność:

>>> import subprocess

>>> proc = subprocess.run(['ls', '-l'])

total 20

drwxrwxr-x. 2 user user 54 Jan 28 15:36 bin

drwxr-xr-x. 2 user user 6 Jan 7 2015 Desktop

-rw-rw-r--. 1 user user 44 Jan 26 22:16 new\_xmen.txt

-rw-rw-r--. 1 user user 98 Jan 26 21:39 read\_file.py

-rw-rw-r--. 1 user user 431 Aug 6 2015 VNCHOWTO

-rw-rw-r--. 1 user user 61 Jan 28 14:11 xmen\_base.txt

-rw-------. 1 user user 68 Mar 18 2016 xrdp-chansrv.log

>>> proc

CompletedProcess(args=['ls', '-l'], returncode=0)

Nasza zmienna proc jest obiektem CompletedProcess, co zapewnia nam dużą elastyczność. Mamy dostęp do atrybutu returncode w naszej zmiennej proc, aby upewnić się, że się udało i zwrócił nam 0. Zwróć uwagę, że komenda ls została wykonana i wydrukowana na ekranie bez określenia, czy chcesz coś wydrukować. Możemy to obejść, przechwytując STDOUT za pomocą podprocesu.PIPE.

>>> proc = subprocess.run(

... ['ls', '-l'],

... stdout=subprocess.PIPE,

... stderr=subprocess.PIPE,

... )

>>> proc

CompletedProcess(args=['ls', '-l'], returncode=0, stdout=b'total 20\ndrwxrwxr-x. 2 user user 54 Jan 28 15:36 bin\ndrwxr-xr-x. 2 user user 6 Jan 7 2015 Desktop\n-rw-rw-r--. 1 user user 44 Jan 26 22:16 new\_xmen.txt\n-rw-rw-r--. 1 user user 98 Jan 26 21:39 read\_file.py\n-rw-rw-r--. 1 user user 431 Aug 6 2015 VNCHOWTO\n-rw-rw-r--. 1 user user 61 Jan 28 14:11 xmen\_base.txt\n-rw-------. 1 user user 68 Mar 18 2016 xrdp-chansrv.log\n', stderr=b'')

>>> proc.stdout

b'total 20\ndrwxrwxr-x. 2 user user 54 Jan 28 15:36 bin\ndrwxr-xr-x. 2 user user 6 Jan 7 2015 Desktop\n-rw-rw-r--. 1 user user 44 Jan 26 22:16 new\_xmen.txt\n-rw-rw-r--. 1 user user 98 Jan 26 21:39 read\_file.py\n-rw-rw-r--. 1 user user 431 Aug 6 2015 VNCHOWTO\n-rw-rw-r--. 1 user user 61 Jan 28 14:11 xmen\_base.txt\n-rw-------. 1 user user 68 Mar 18 2016 xrdp-chansrv.log\n'

Teraz, gdy przechwyciliśmy dane wyjściowe do atrybutów w naszej zmiennej proc, możemy z nimi pracować w naszym skrypcie i ustalić, czy kiedykolwiek powinien zostać wydrukowany. Spójrz na ten ciąg poprzedzony znakiem b. Dzieje się tak, ponieważ jest to obiekt bajtowy, a nie łańcuch. Typ bajtów może zawierać tylko znaki ASCII i po wydrukowaniu nie zrobi nic specjalnego z sekwencjami ucieczki. Jeśli chcemy wykorzystać tę wartość jako łańcuch, musimy jawnie przekonwertować ją za pomocą metody bytes.decode.

>>> print(proc.stdout)

b'total 20\ndrwxrwxr-x. 2 user user 54 Jan 28 15:36 bin\ndrwxr-xr-x. 2 user user 6 Jan 7 2015 Desktop\n-rw-rw-r--. 1 user user 44 Jan 26 22:16 new\_xmen.txt\n-rw-rw-r--. 1 user user 98 Jan 26 21:39 read\_file.py\n-rw-rw-r--. 1 user user 431 Aug 6 2015 VNCHOWTO\n-rw-rw-r--. 1 user user 61 Jan 28 14:11 xmen\_base.txt\n-rw-------. 1 user user 68 Mar 18 2016 xrdp-chansrv.log\n'

>>> print(proc.stdout.decode())

total 20

drwxrwxr-x. 2 user user 54 Jan 28 15:36 bin

drwxr-xr-x. 2 user user 6 Jan 7 2015 Desktop

-rw-rw-r--. 1 user user 44 Jan 26 22:16 new\_xmen.txt

-rw-rw-r--. 1 user user 98 Jan 26 21:39 read\_file.py

-rw-rw-r--. 1 user user 431 Aug 6 2015 VNCHOWTO

-rw-rw-r--. 1 user user 61 Jan 28 14:11 xmen\_base.txt

-rw-------. 1 user user 68 Mar 18 2016 xrdp-chansrv.log

>>>

**Intentionally Raising Errors**

Funkcja subprocess.run domyślnie nie zgłosi błędu, jeśli wykonasz coś, co zwraca niezerowy kod wyjścia. Oto przykład tego:

>>> new\_proc = subprocess.run(['cat', 'fake.txt'])

cat: fake.txt: No such file or directory

>>> new\_proc

CompletedProcess(args=['cat', 'fake.txt'], returncode=1)

W tej sytuacji możemy chcieć zgłosić błąd i jeśli przekażemy argument check do funkcji, podniesie subprocess.CalledProcessError, jeśli coś pójdzie nie tak:

>>> error\_proc = subprocess.run(['cat', 'fake.txt'], check=True)

cat: fake.txt: No such file or directory

Traceback (most recent call last):

File "", line 1, in

File "/usr/local/lib/python3.6/subprocess.py", line 418, in run

output=stdout, stderr=stderr)

subprocess.CalledProcessError: Command '['cat', 'fake.txt']' returned non-zero exit status 1.

>>>