Part 84
-
Multi-Tasking
-
SubProcesses

Version: 2021-01-14

Subprocesses

Subprocesses were introduced to replace the following various old modules (functions) present in Python:

- os.system
- os.spawn
- os.popen
- commands

Subprocesses are used to run independent programs (or commands).

Subprocesses `call`

This function is used to run a command and get its return code. An example of its use

```
import subprocess
print(subprocess.call(["date"]))
```

Outputs this

```
Sun 24 May 18:12:00 BST 2020
0
```

Parameters can be passed to the command as follows

```
import subprocess
print(subprocess.call(["pwd", "-P"]))
```

Outputs this

```
/home/pi
```

Subprocess run

This function is like the call method and it runs a command. Additionally, the return code of the command is also displayed. An example of its use

```
import subprocess
print(subprocess.run(["pwd", "-P"]))
```

Outputs this

```
/home/pi
ComletedProcess(args='pwd', '-P'], returncode=0)
```

Subprocess check_call

This function is like the `call` method and it runs a command. Additionally, if there was an error in running the specified command, it raises an exception (`CalledProcessError`). An example of its use:

```
print(subprocess.check_call("false"))
```

Outputs this

```
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "/usr/lib/python3.7/subprocess.py", line 347, in check_call
Subprocess.CalledProcessError: Command 'false' returned non-zero exit status
1.
```

Subprocess check_output
The output is bound to the parent process and is not accessible when we use the `call` function to run a command. The `check_output` function can be used to capture the output as shown below

```
import subprocess
print(subprocess.check_output(["pwd", "-P"]))
```

Outputs this

```
b'/home/pi\n'
```

Subprocess Popen and communicate
`Popen` is used to execute a child program in a new process. The `communicate` function is used to read input and output from the process itself, where `stdout` and `stderr` are the process output and input respectively. An example is shown below:

```
import subprocess
pr = subprocess.Popen(["pwd", "-P"], stdout=subprocess.PIPE)
stdout, stderr = pr.communicate()
print(stdout)
```

Outputs this

```
b'/home/pi\n'
```

Running a Python program
An example is given to show how a Python program can be run using the `subprocess` module. Program `hello.py` contains the following line:

```
nano hello.py

print("Hello from program")
```

Save and exit. In addition, create program `disp.py`:

```
nano disp.py

import subprocess
subprocess.call(["python3", "hello.py"])
```

Save and exit. Run the program `disp.py` to get

```
python3 disp.py
Hello from program
```

Note: that subprocess `call` is blocking, but we can use `Popen` to execute a child process so that the parent process is not blocked.