return value from one python script to another

Asked 8 years, 8 months ago   Modified 8 years, 8 months ago   Viewed 60k times

▲

13

▼

🔖

🕑

I have two files: script1.py and script2.py. I need to invoke script2.py from script1.py and return the value from script2.py back to script1.py. But the catch is script1.py actually runs script2.py through os.

script1.py:

```
import os
print(os.system("script2.py 34"))
```

script2.py

```
import sys
def main():
    x="Hello World"+str(sys.argv[1])
    return x

if __name__ == "__main__":
    x= main()
```

As you can see, I am able to get the value into script2, but not back to script1. How can I do that? NOTE: script2.py HAS to be called as if its a commandline execution. Thats why I am using os.

python   python-3.x   os.system

Share Improve this question Follow

asked Jun 5, 2015 at 10:27

codingsplash
4,905 🟡 12 ⚪ 52 🟤 92

What should be returned from `script2.py` ? Is return code sufficient? – Łukasz Rogalski Jun 5, 2015 at 10:33 ✏️

I want the value x to be returned. In this case script2 should return "Hello World 34"
– codingsplash Jun 5, 2015 at 10:35

---

2 Answers

Sorted by: Highest score (default) ⇕

Ok, if I understand you correctly you want to:

20

- pass an argument to another script
- retrieve an output from another script to original caller

I'll recommend using subprocess module. Easiest way would be to use <u>check_output()</u> function.

> Run command with arguments and return its output as a byte string.

Sample solution:

script1.py

```
import sys
import subprocess
s2_out = subprocess.check_output([sys.executable, "script2.py", "34"])
print s2_out
```

script2.py:

```
import sys
def main(arg):
    print("Hello World"+arg)

if __name__ == "__main__":
    main(sys.argv[1])
```

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

The recommended way to return a value from one python "script" to another is to import the script as a Python module and call the functions directly:

12

```python
import another_module

value = another_module.get_value(34)
```

where `another_module.py` is:

```python
#!/usr/bin/env python
def get_value(*args):
    return "Hello World " + ":".join(map(str, args))

def main(argv):
    print(get_value(*argv[1:]))

if __name__ == "__main__":
    import sys
    main(sys.argv)
```

You could both import `another_module` and run it as a script from the command-line. If you don't need to run it as a command-line script then you could remove `main()` function and `if __name__ == "__main__"` block.

See also, [Call python script with input with in a python script using subprocess](#).

Share Improve this answer

Follow

edited May 23, 2017 at 10:30

Community `Bot`
1 • 1

answered Jun 5, 2015 at 12:02

jfs
405k • 199 • 1012 • 1693