Python Programming

Circular Imports 2

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Same last codes but check in sys

a.py

```
1  def check(m):
2      import sys
3      return str(m in sys.modules)
4
5  print(f"in top a.by: a {check('a')}")
6  print(f"in top a.by: b {check('b')}")
7
8  import b
9  def af():
10     return b.x
11 af()
12
13 print(f"in bottom a.by: a {check('a')}")
14 print(f"in bottom a.by: b {check('b')}")
15
```

b.py

```
1  def check(m):
2      import sys
3      return str(m in sys.modules)
4
5  print(f"in top b.by: a {check('a')}")
6  print(f"in top b.by: b {check('b')}")
7
8  import a
9  x = 1
10 def bf():
11     print(a.af())
12
13 print(f"in bottom b.by: a {check('a')}")
14 print(f"in bottom b.by: b {check('b')}")
```

Let’s run d.py

a.py

```
1  import b
2  def af():
3      return b.x
4  af()
```

b.py

```
1  import a
2  x = 1
3  def bf():
4      print(a.af())
```

c.py

```
1  import a
2
```

a.py

```
1   def check(m):
2       import sys
3       return str(m in sys.modules)
4
5   print(f"in top a.by: a {check('a')}")
6   print(f"in top a.by: b {check('b')}")
7
8   import b
9   def af():
10      return b.x
11  af()
12
13  print(f"in bottom a.by: a {check('a')}")
14  print(f"in bottom a.by: b {check('b')}")
15
```

b.py

```
1   def check(m):
2       import sys
3       return str(m in sys.modules)
4
5   print(f"in top b.by: a {check('a')}")
6   print(f"in top b.by: b {check('b')}")
7
8   import a
9   x = 1
10  def bf():
11      print(a.af())
12
13  print(f"in bottom b.by: a {check('a')}")
14  print(f"in bottom b.by: b {check('b')}")
```

```
in top a.by: a True
in top a.by: b False
in top b.by: a True
in top b.by: b True
in bottom b.by: a True
in bottom b.by: b True
in bottom a.by: a True
in bottom a.by: b True
```

Let’s run d.py

a.py

```
1  import b
2  def af():
3      return b.x
4  af()
```

b.py

```
1  import a
2  x = 1
3  def bf():
4      print(a.af())
```

d.py

```
1  import b
2
```

a.py

```
1  def check(m):
2      import sys
3      return str(m in sys.modules)
4
5  print(f"in top a.by: a {check('a')}")
6  print(f"in top a.by: b {check('b')}")
7
8  import b
9  def af():
10     return b.x
11 af()
12
13 print(f"in bottom a.by: a {check('a')}")
14 print(f"in bottom a.by: b {check('b')}")
15
```

b.py

```
1  def check(m):
2      import sys
3      return str(m in sys.modules)
4
5  print(f"in top b.by: a {check('a')}")
6  print(f"in top b.by: b {check('b')}")
7
8  import a
9  x = 1
10 def bf():
11     print(a.af())
12
13 print(f"in bottom b.by: a {check('a')}")
14 print(f"in bottom b.by: b {check('b')}")
```

```
in top b.by: a False
in top b.by: b True
in top a.by: a True
in top a.by: b True
Traceback (most recent ca
  File "/home/moustafa/00
    import b
  File "/home/moustafa/co
    import a
  File "/home/moustafa/co
    af()
  File "/home/moustafa/co
    return b.x
AttributeError: partially
```

Let's run b.py

a.py

```
1  import b
2  def af():
3      return b.x
4  af()
```

b.py

```
1  import a
2  x = 1
3  def bf():
4      print(a.af())
```

a.py

```
1  def check(m):
2      import sys
3      return str(m in sys.modules)
4
5  print(f"in top a.by: a {check('a')}")
6  print(f"in top a.by: b {check('b')}")
7
8  import b
9  def af():
10     return b.x
11 af()
12
13 print(f"in bottom a.by: a {check('a')}")
14 print(f"in bottom a.by: b {check('b')}")
15
```

b.py

```
1  def check(m):
2      import sys
3      return str(m in sys.modules)
4
5  print(f"in top b.by: a {check('a')}")
6  print(f"in top b.by: b {check('b')}")
7
8  import a
9  x = 1
10 def bf():
11     print(a.af())
12
13 print(f"in bottom b.by: a {check('a')}")
14 print(f"in bottom b.by: b {check('b')}")
```

```
in top b.by: a False
in top b.by: b False
in top a.by: a True
in top a.by: b False
in top b.by: a True
in top b.by: b True
in bottom b.by: a True
in bottom b.by: b True
in bottom a.by: a True
in bottom a.by: b True
in bottom b.by: a True
in bottom b.by: b True
```

Handling Cycles

1. This is a bad design. Redesign it
2. Otherwise: consider the following **workarounds**
3. Merge files together if makes sense (try to respect single responsibility)
4. Delay imports as possible if that breaks cycle (e.g. move inside function)
5. Use TYPE_CHECKING / Conditional imports

dir function

```
print(dir())    # dir() returns a list of defined names in a namespace
# ['__annotations__', '__builtins__', '__cached__', '__doc__', '__file__',

import ourlib
print(dir())
# ['__annotations__', '__builtins__', ... 'ourlib']

print(dir(ourlib))
# [, 'sq', 'sum1n']

from ourlib import sq
print(dir())
# ['__annotations__', '__builtins__', ... 'ourlib', 'sq']

from ourlib import *
print(dir())
# ['__annotations__', '__builtins__', ... 'ourlib', 'sq', 'sum1n']
```

Reloading

- Sometimes we may want to reload a module during running
- E.g. it was updated during running
  - import **importlib**
  - importlib.reload(**my_module**)

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."