Why is a class variable not accessible from a method?

Asked 8 years ago Modified 11 months ago Viewed 2k times

6

Consider:

```
class Foo:
    a = 1
    def bar():
        print(a)
```

I expected `a` to be available to the method by scoping rules: local first, then enclosing, ...

The class `Foo` creates a namespace and a scope, does it not?

`bar` creates a scope; isn't it enclosed by the scope of the class? There is no `a` defined in the scope of `bar`, so I expected it to pick up the class variable from the enclosing scope.

Evidently I'm confused about namespaces and scopes. I've tried reading up on this, but haven't been able to find definitive clarification on this particular point (`self.a` works, of course).

python namespaces scope

Share Improve this question Follow

edited Feb 15, 2022 at 1:20

mkrieger1
16.9k 4 51 58

asked Jan 23, 2015 at 22:05

garyp
949 7 34

Try `Foo.a` instead of `a`. – Simpom Jan 23, 2015 at 22:07

2 Answers

Sorted by:

Highest score (default)

9

The class body is not a nestable scope, no. The Python *Execution Model* explicitly excludes it:

> The scope of names defined in a class block is limited to the class block; it does not extend to the code blocks of methods

That's because the body of a class is executed to form the class attributes; see it as a function with locals, and the locals become the attributes of the new class object.

You can then access those attributes either on the class directly (`Foo.a`) or via an instance (where attribute lookup falls through to the class).

Share Improve this answer Follow

answered Jan 23, 2015 at 22:07

Martijn Pieters ♦
1.0m 283 3973 3296

-1

```
class Foo:
    def __init__(self):
        self.a = 1

    def bar(self):
        print(self.a)
```

By using `__init__()` you will be able to pass the variables to other functions.

Share Improve this answer Follow

answered Jan 23, 2015 at 22:11

Jordy19
106 2 11

1 This isn't what the OP is asking about. The question is *why* names at the class scope are not accessed in methods, like in nested function scopes. – Martijn Pieters ♦ Jan 23, 2015 at 22:14