Class inheritance in Python, super and overridden methods

Asked 1 year, 10 months ago Active 1 year, 10 months ago Viewed 86 times

1

Consider the following code:

```
class A:
    def foo(self, a):
        return a
    def bar(self, a):
        print(foo(a))

class B(A):
   def foo(self, a):
       return a[0]
```

Now calling `B.bar(a)` the result is `print(a[0])`, but what I want is `print(a)`. More directly: I'd like that the calling of `bar()` in a child class uses the definition of `foo` given in `A` even if overridden. How do i do that?

python inheritance overriding self Edit tags

edited Dec 26 '18 at 18:19

Jonah Bishop
11.3k 4 36 65

asked Dec 26 '18 at 18:18

L.A.
137 7

Possible duplicate of Python class inherits object – MilaDroid Dec 26 '18 at 18:24

I can't find the answer to my question... – L.A. Dec 26 '18 at 18:39

1 Answer

Active | Oldest | Votes

0

I believe this is what you are looking for:

```
class A(object):
    def foo(self, a):
        return a
    def bar(self, a):
        print(A.foo(self,a))

class B(A):
   def foo(self, a):
       return a[0]
```

or alternatively:

```
class A:
    def foo(self, a):
        return a
```

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

class B(A):
    def foo(self, a):
        return super().foo(a)
```

answered Dec 26 '18 at 18:42

gold_cy
7,930 2 12 35

Perfect, the first one you written is working as I want! Thanks – L.A. Dec 26 '18 at 18:49