

Informatique 1
Introduction à la programmation

Mission 10 : restructuration

Kim Mens

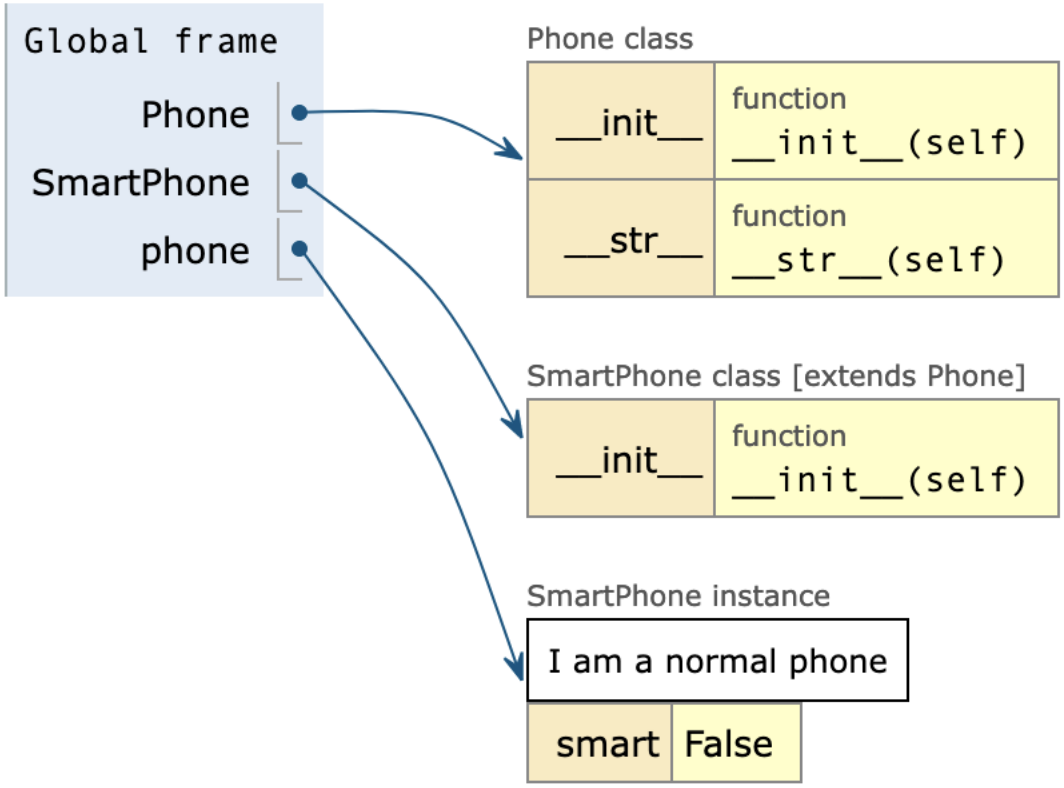
La portée des variables

- variable d’instance
  - publique
  - privée

La portée d'une variable d'instance publique

```
class Phone :
  def __init__(self):
    self.smart = False
  def __str__(self):
    if self.smart :
      return 'I am a smartphone'
    else :
      return 'I am a normal phone'

class SmartPhone(Phone):
  def __init__(self):
    self.smart = True
    super().__init__()
```

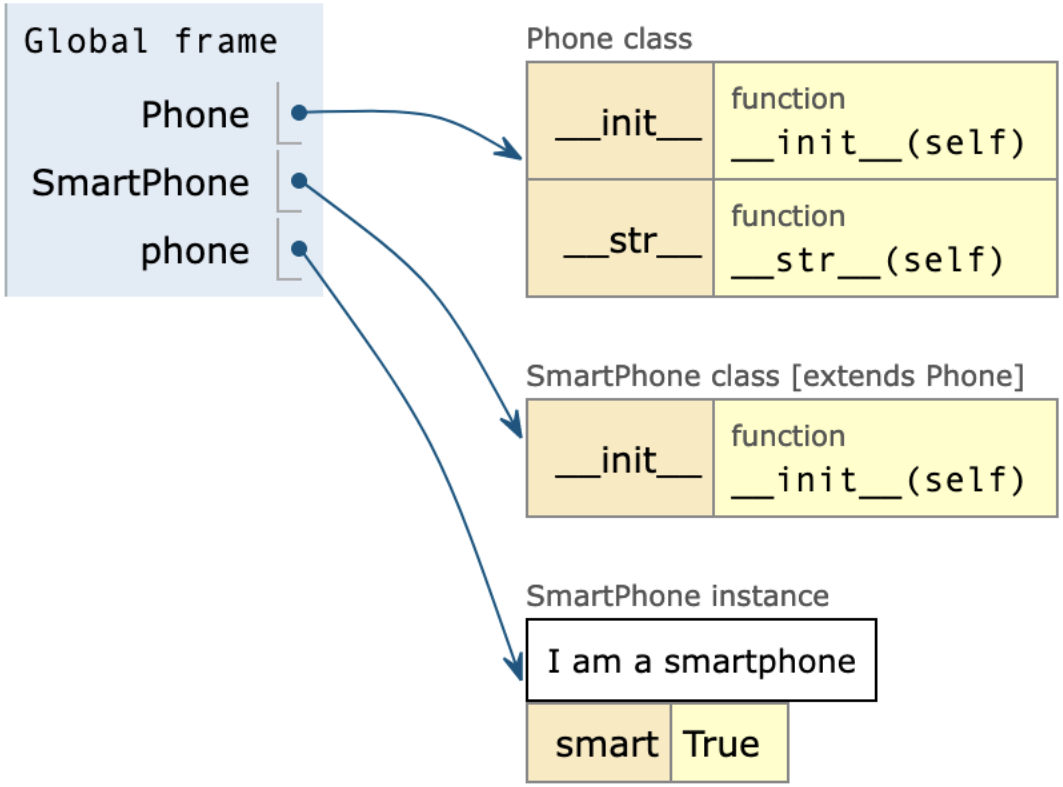



```
>>> phone = SmartPhone()
>>> print(phone)
I am a normal phone
```

La portée d'une variable d'instance publique

```
class Phone :
  def __init__(self):
    self.smart = False
  def __str__(self):
    if self.smart :
      return 'I am a smartphone'
    else :
      return 'I am a normal phone'

class SmartPhone(Phone):
  def __init__(self):
    super().__init__()
    self.smart = True
```

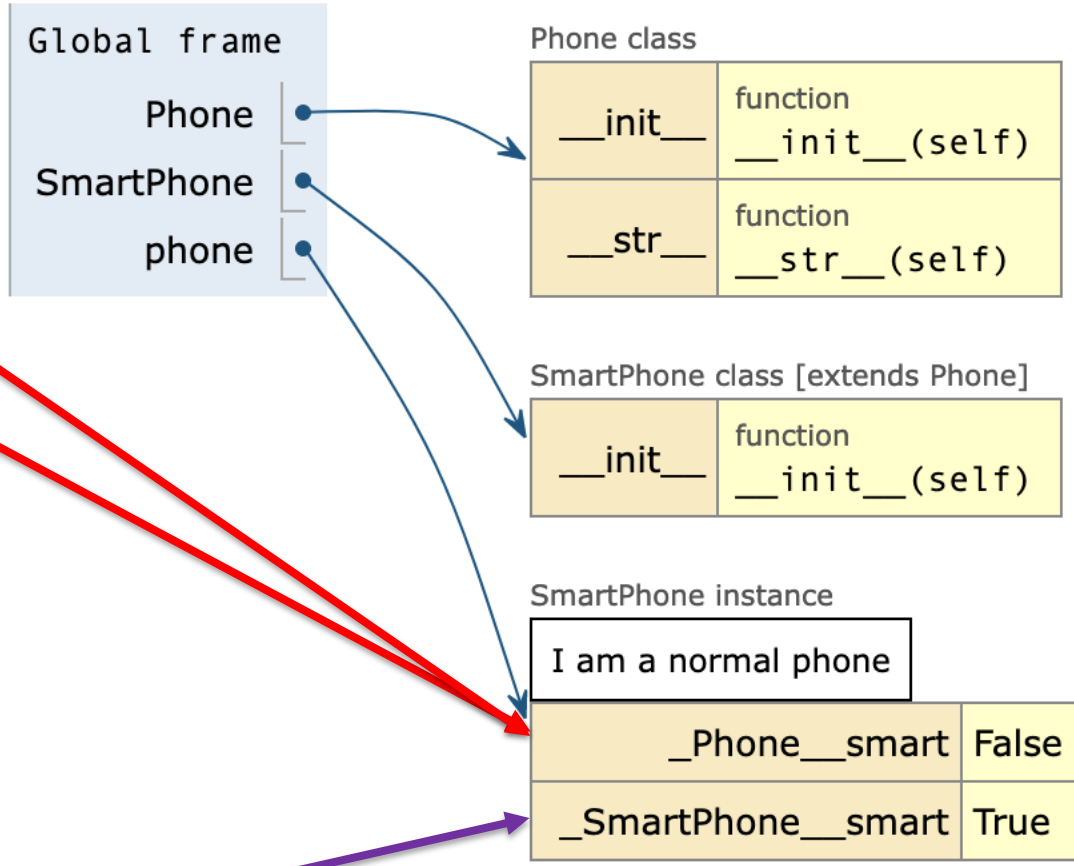



```
>>> phone = SmartPhone()
>>> print(phone)
I am a smartphone
```

La portée d'une variable d'instance privée

```
class Phone :
  def __init__(self):
    self.__smart = False
  def __str__(self):
    if self.__smart :
      return 'I am a smartphone'
    else :
      return 'I am a normal phone'

class SmartPhone(Phone):
  def __init__(self):
    super().__init__()
        self.__smart = True
```

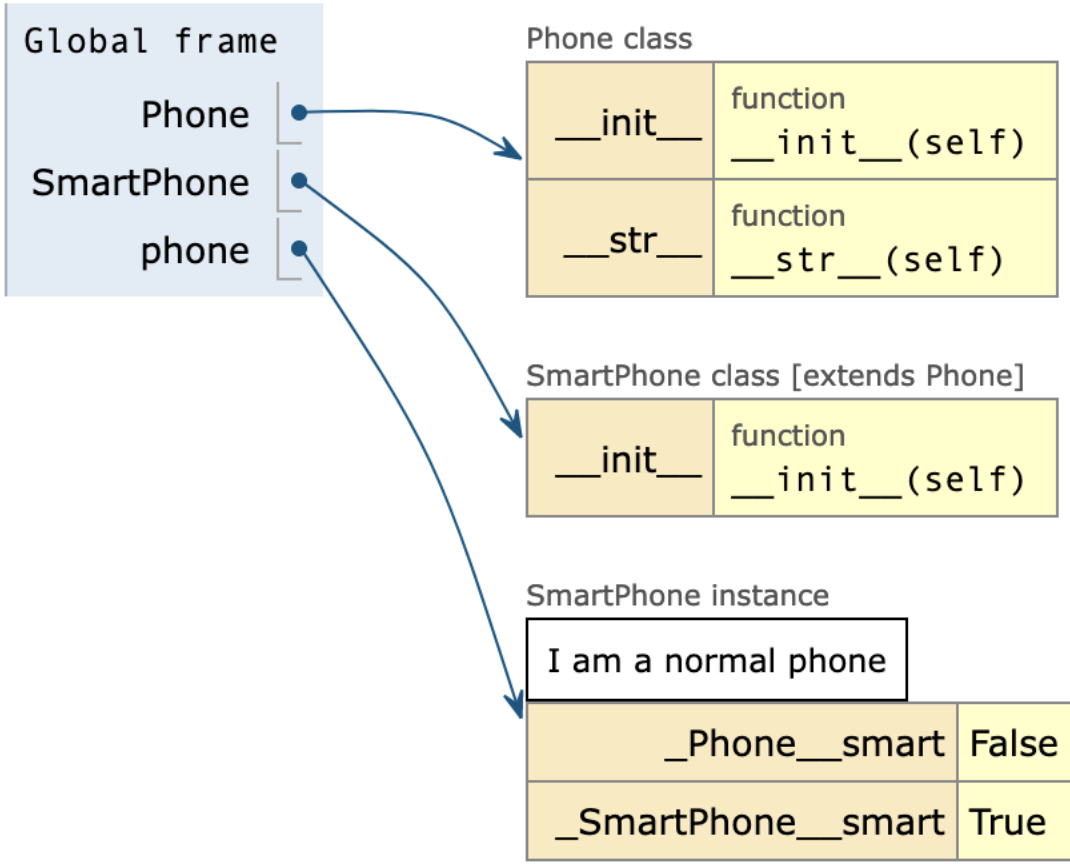



```
>>> phone = SmartPhone()
>>> print(phone)
I am a normal phone
```

La portée d'une variable d'instance privée

```
class Phone :
  def __init__(self):
    self.__smart = False
  def __str__(self):
    return 'I am a normal phone'

class SmartPhone(Phone):
  def __init__(self):
    super().__init__()
        self.__smart = True
  def __str__(self):
    return 'I am a smartphone'
```

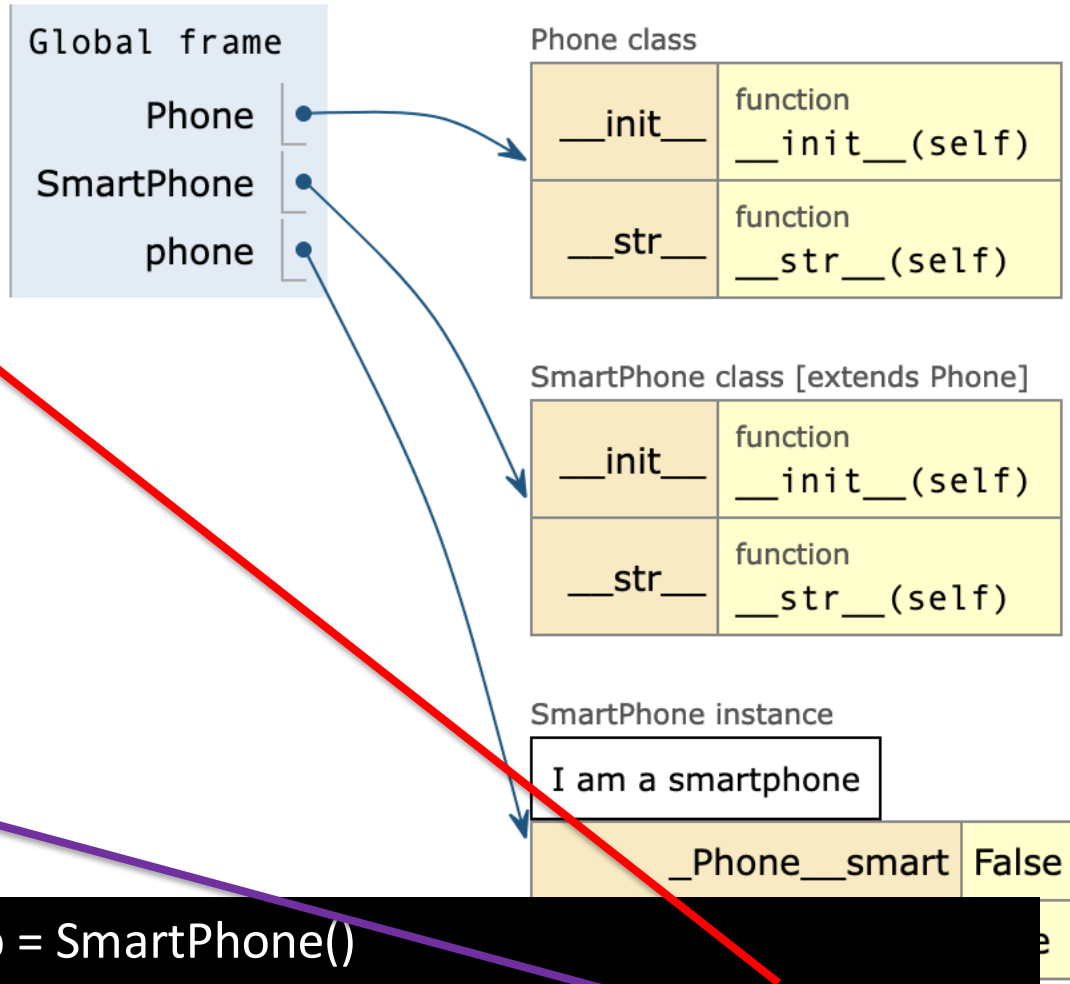



```
>>> phone = SmartPhone()
>>> print(phone)
I am a smartphone
```

La portée d'une variable d'instance privée

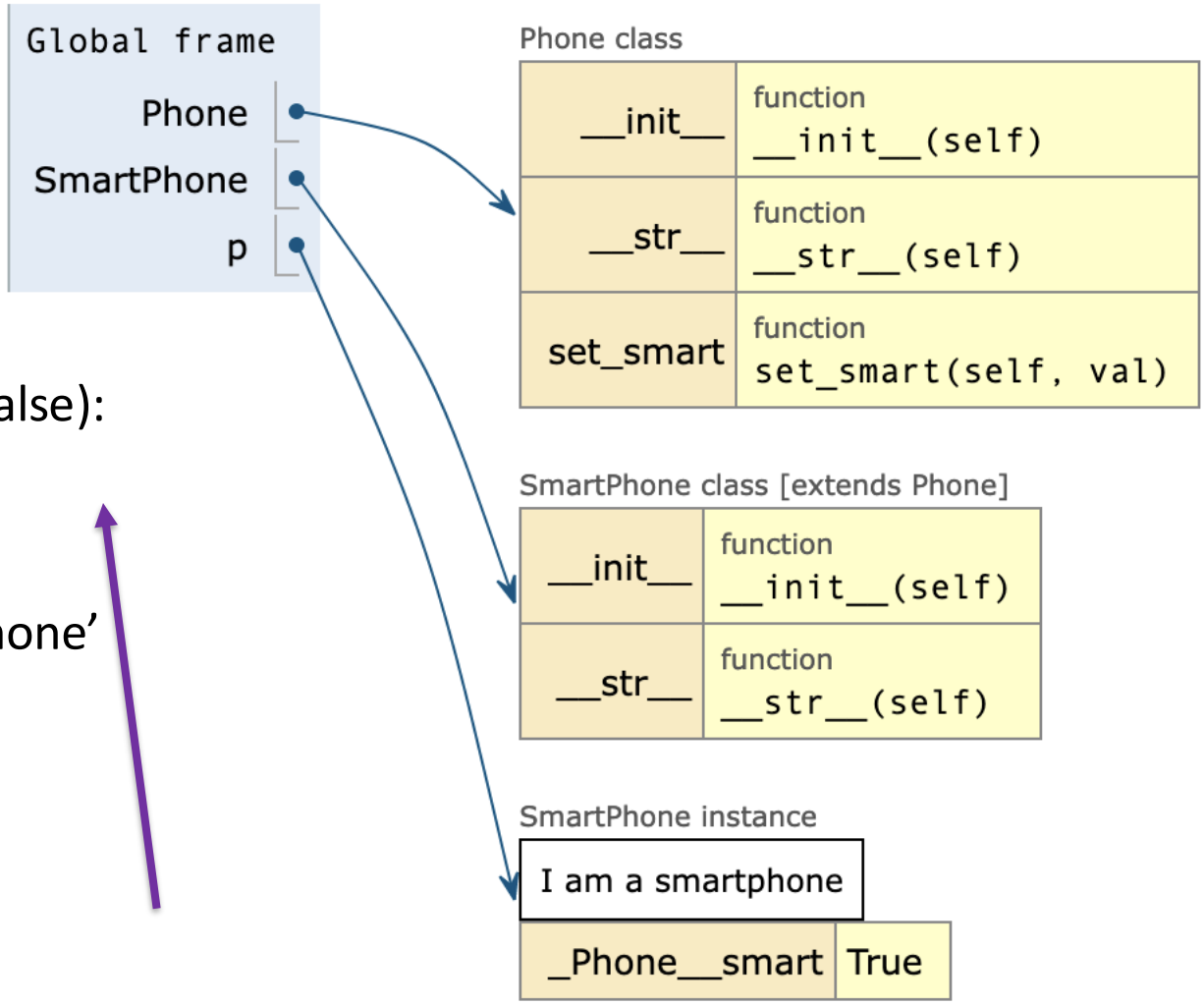
```
class Phone :
  def __init__(self):
    self.__smart = False
  def __str__(self):
    return 'I am a normal phone'

class SmartPhone(Phone):
  def __init__(self):
    super().__init__()
        self.__smart = True
  def __str__(self):
    return 'I am a smartphon
```



```
class Phone :
  def __init__(self,smart=False):
    self.__smart = smart
  def __str__(self):
    return 'I am a normal phone’

class SmartPhone(Phone):
  def __init__(self):
    super().__init__(True)
  def __str__(self):
    return 'I am a smartphone'
```




```
>>> p = SmartPhone()
>>> print(p)
I am a smart phone
>>> print(p._Phone__smart)
True
```

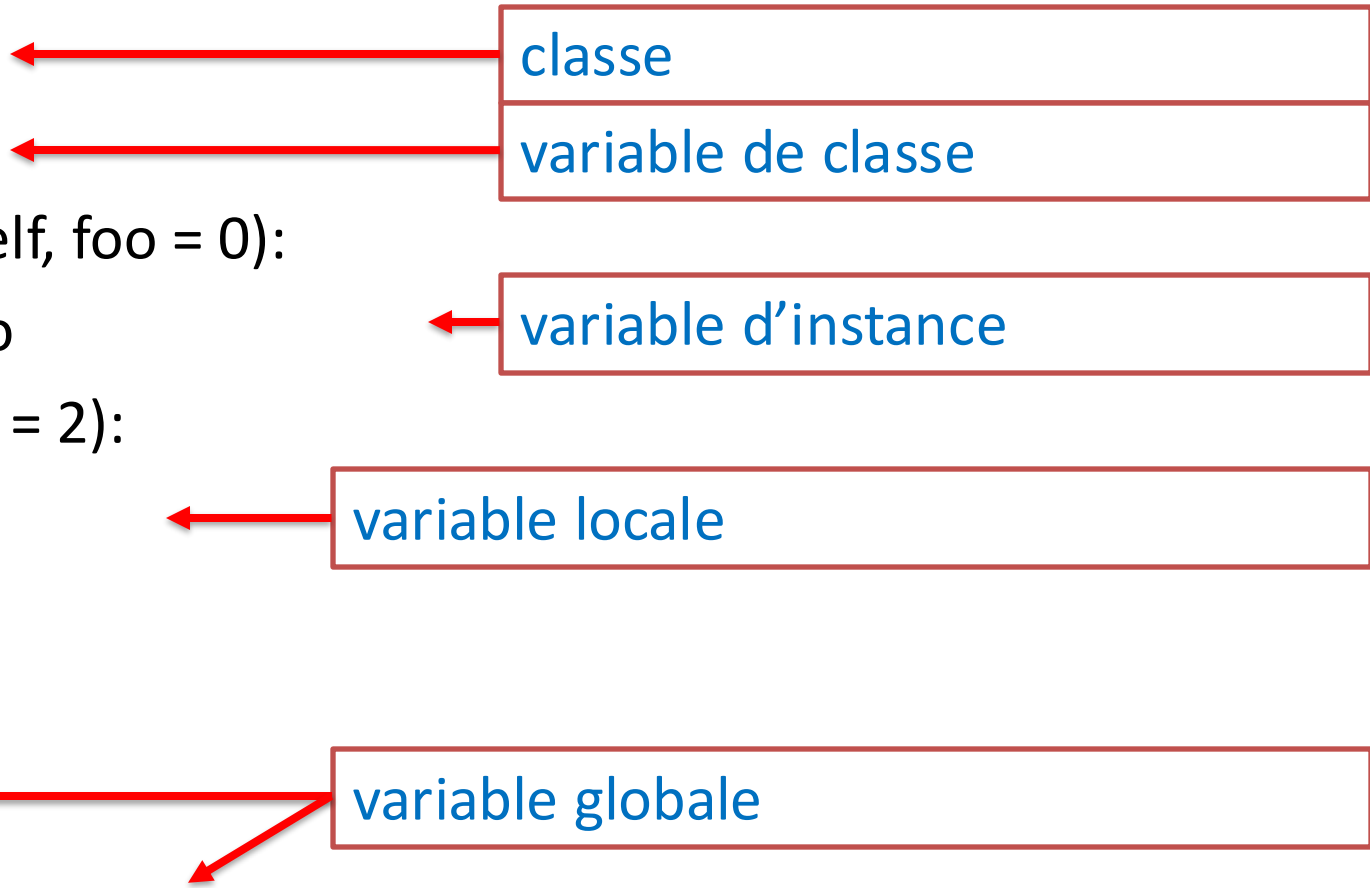
La portée des variables

- classe
- variable d’instance
- variable de classe
- variable locale
- variable globale
- paramètres et arguments

La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self, foo = 2):
        foo = 3
        return foo

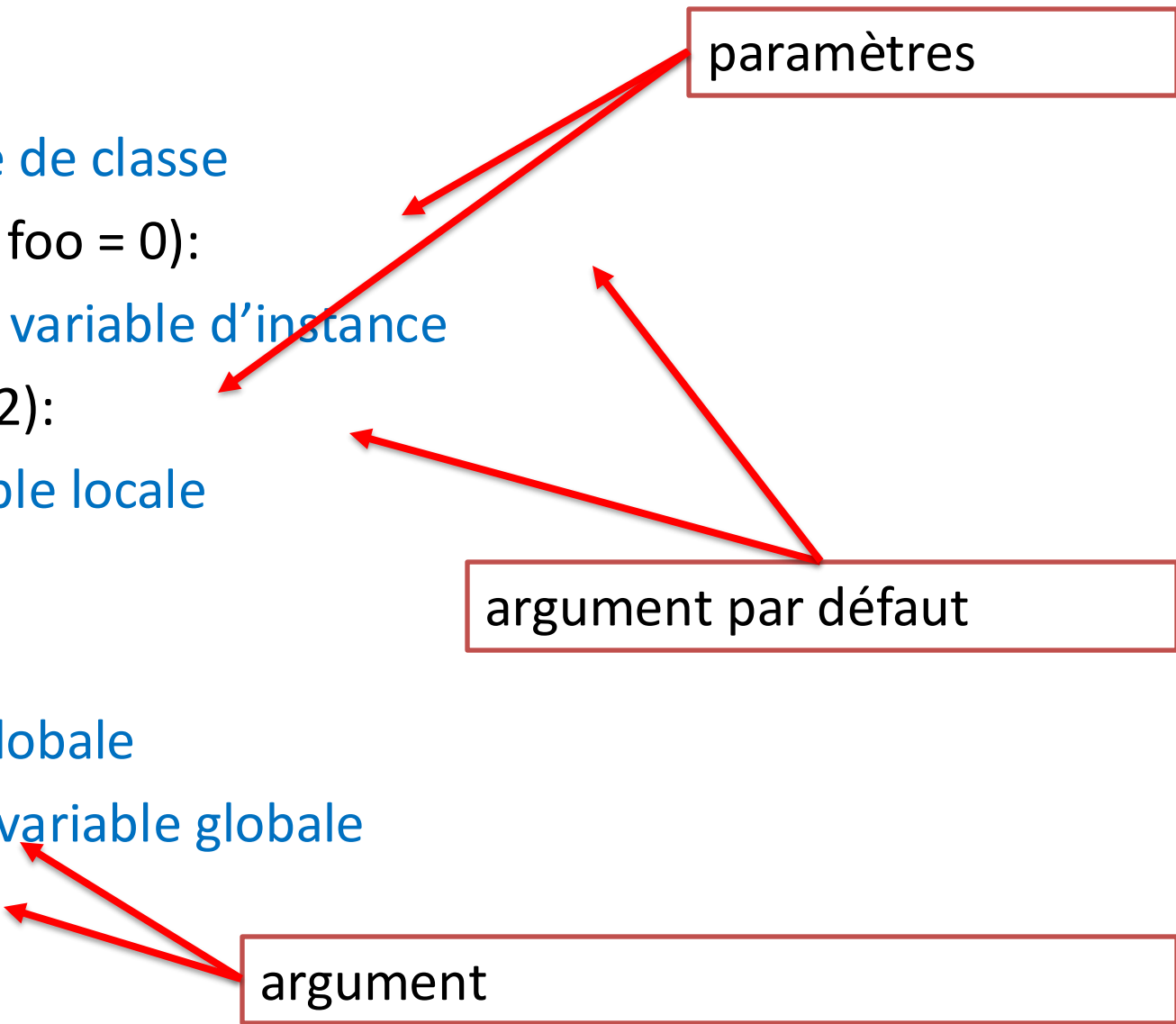
foo = 4
bar = Foo(foo+2)
print(bar.m(5))
```



La portée des variables

```
class Foo : # classe
    foo = 1 # variable de classe
    def __init__(self, foo = 0):
        self.foo = foo # variable d'instance
    def m(self, foo = 2):
        foo = 3 # variable locale
        return foo

foo = 4 # variable globale
bar = Foo(foo+2) # variable globale
print(bar.m(5))
```

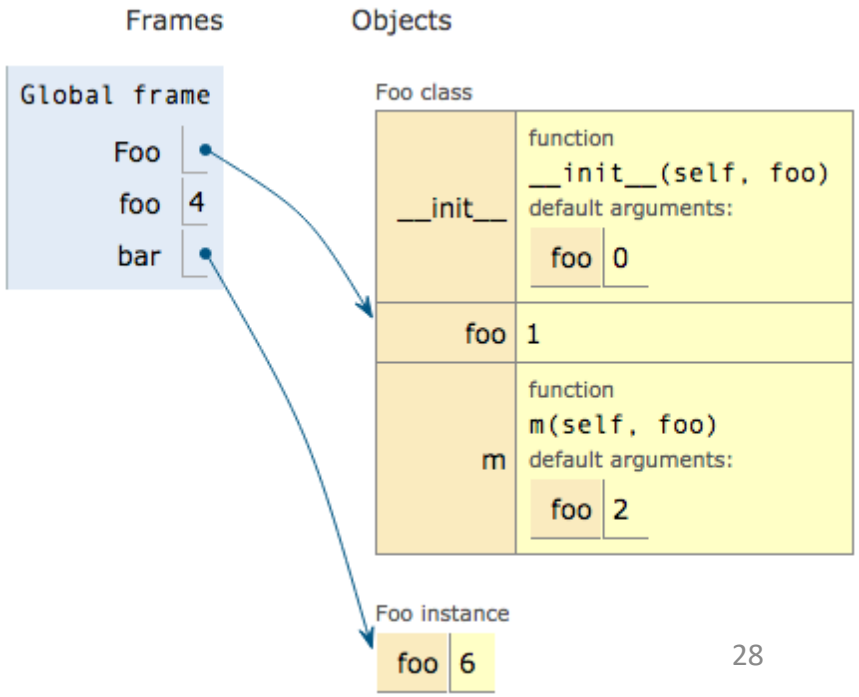


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self, foo = 2):
        foo = 3
        return foo

foo = 4
bar = Foo(foo+2)
print(bar.m(5))
```

?

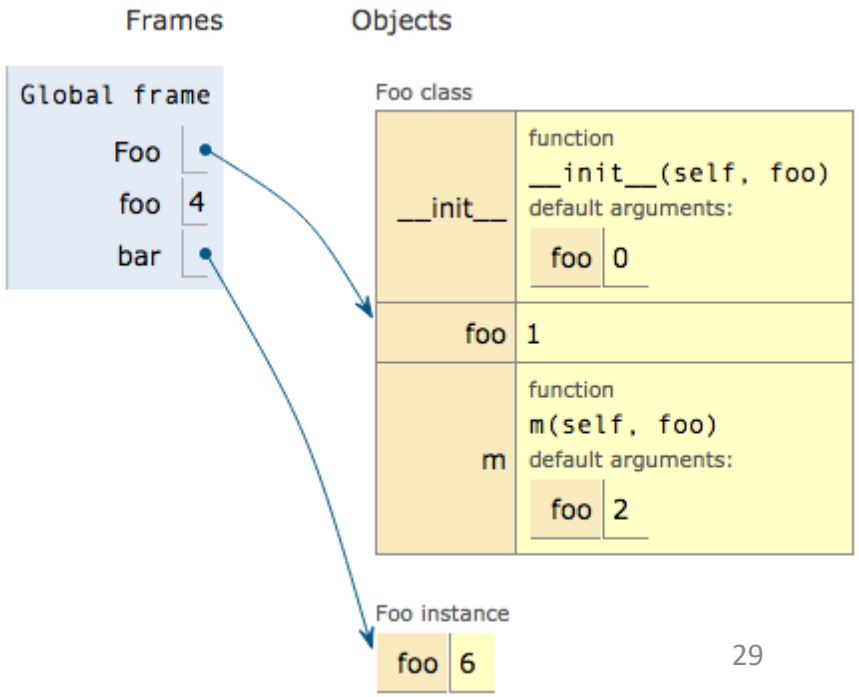


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self, foo = 2):
        foo = 3
        return foo

foo = 4
bar = Foo(foo+2)
print(bar.m(5))
```

3

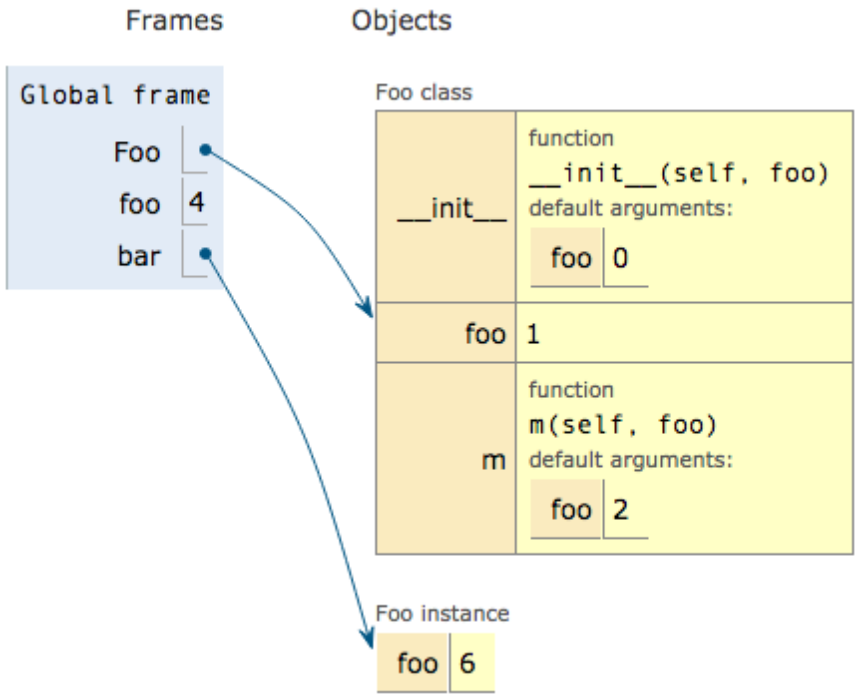


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self, foo = 2):
        #foo = 3
        return foo

foo = 4
bar = Foo(foo+2)
print(bar.m(5))
```

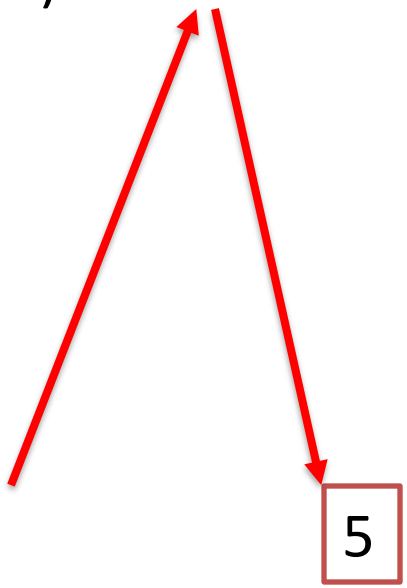
?

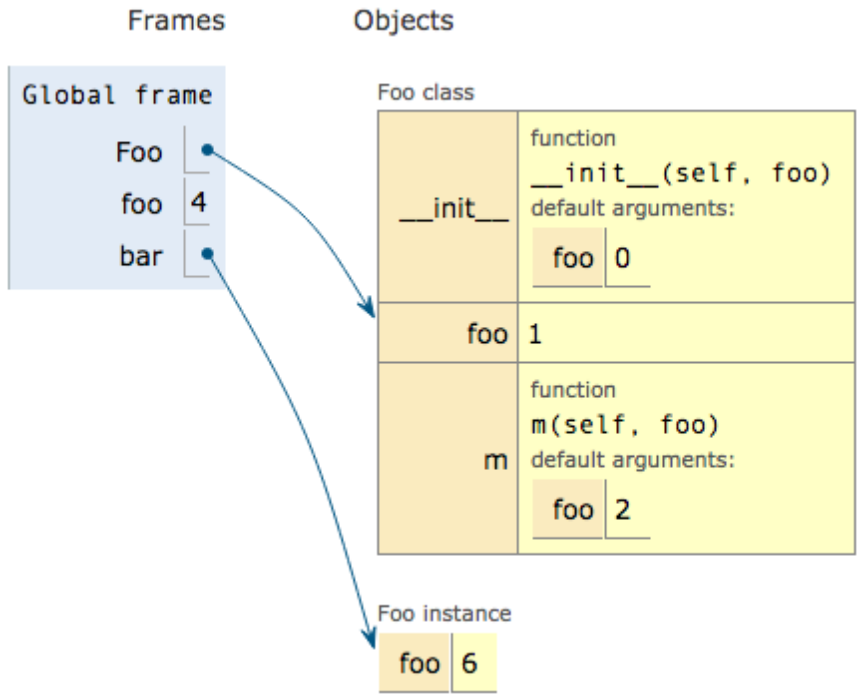


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self, foo = 2):
        return foo

foo = 4
bar = Foo(foo+2)
print(bar.m(5))
```



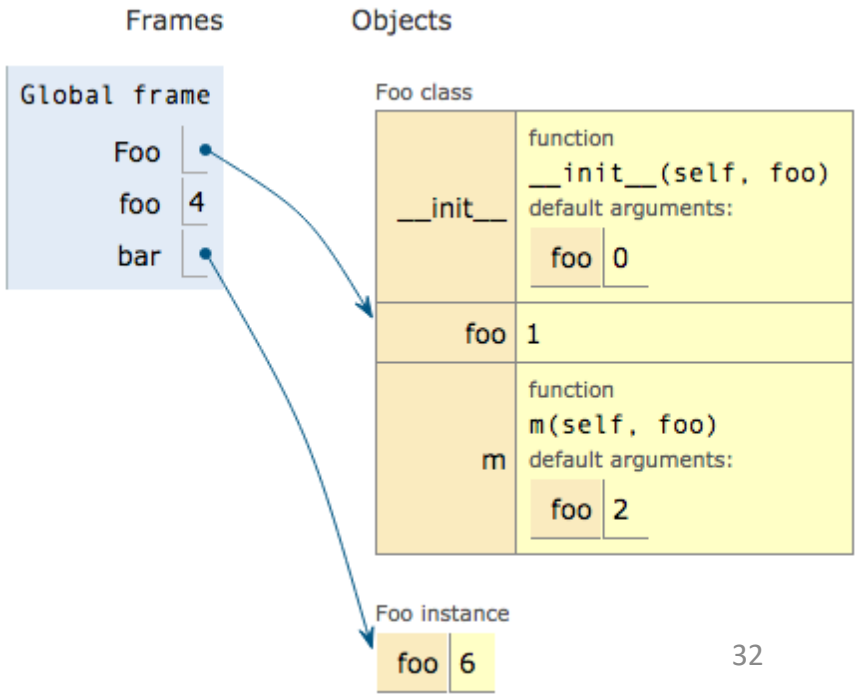


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self, foo = 2):
        return foo

foo = 4
bar = Foo(foo+2)
print(bar.m())
```

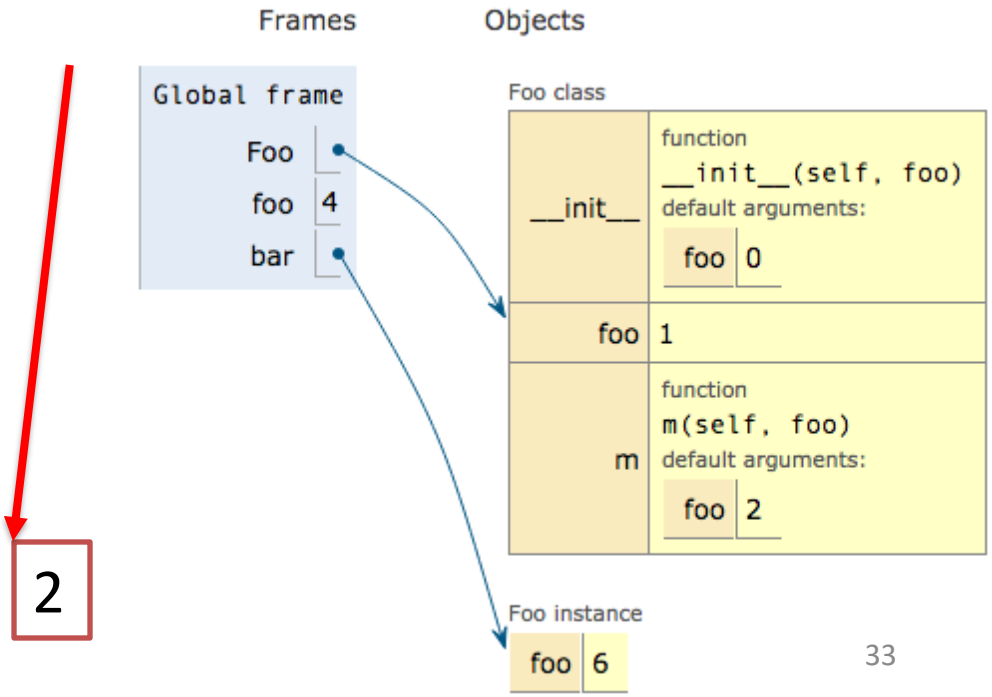
?



La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self, foo = 2):
        return foo


foo = 4
bar = Foo(foo+2)
print(bar.m())
```

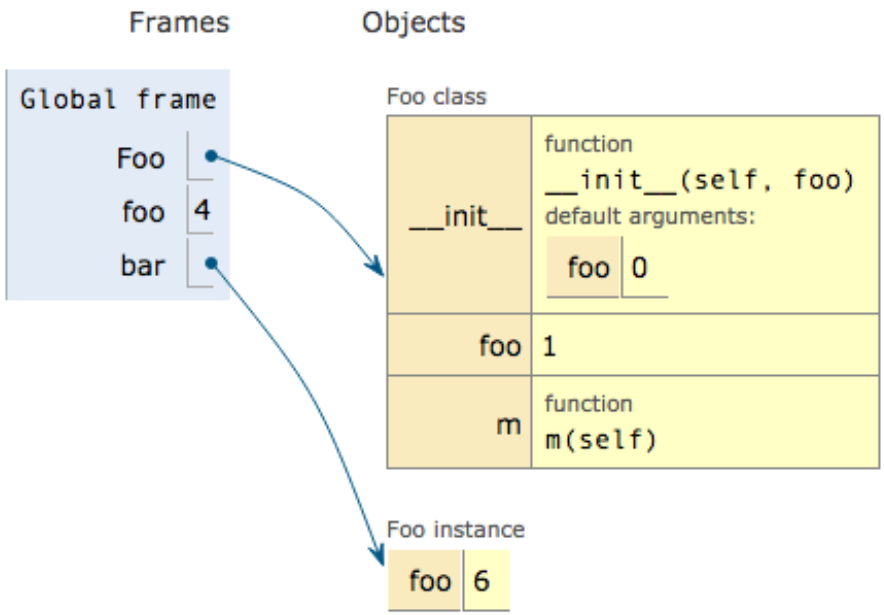


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self, foo = 2):
        return foo


foo = 4
bar = Foo(foo+2)
print(bar.m())
```

?

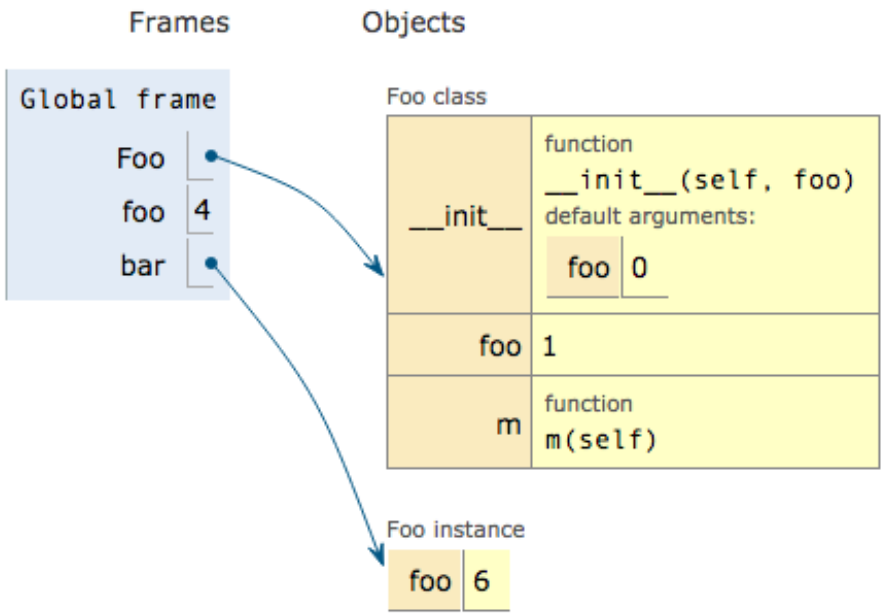


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self):
        return foo

foo = 4
bar = Foo(foo+2)
print(bar.m())
```

4

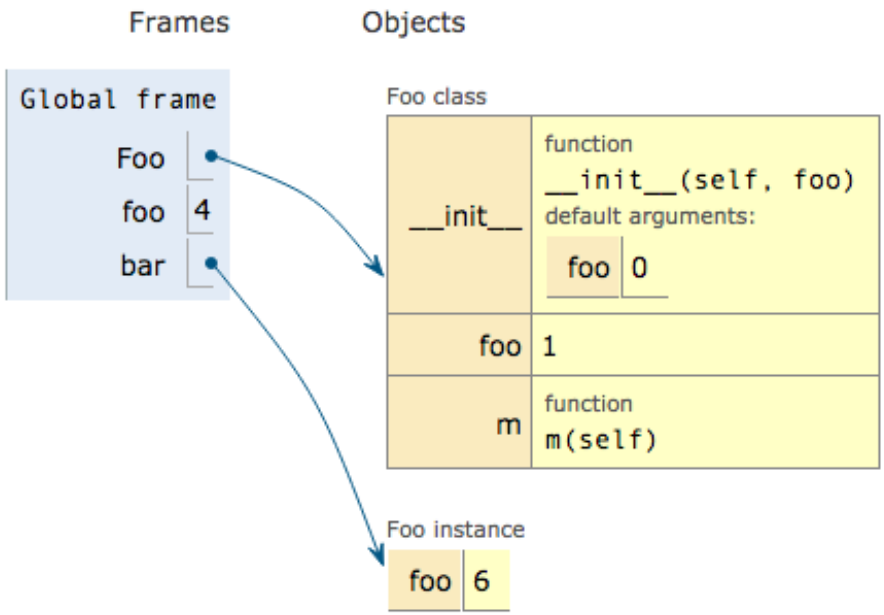


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self):
        return self.foo
```

```
foo = 4
bar = Foo(foo+2)
print(bar.m())
```

?

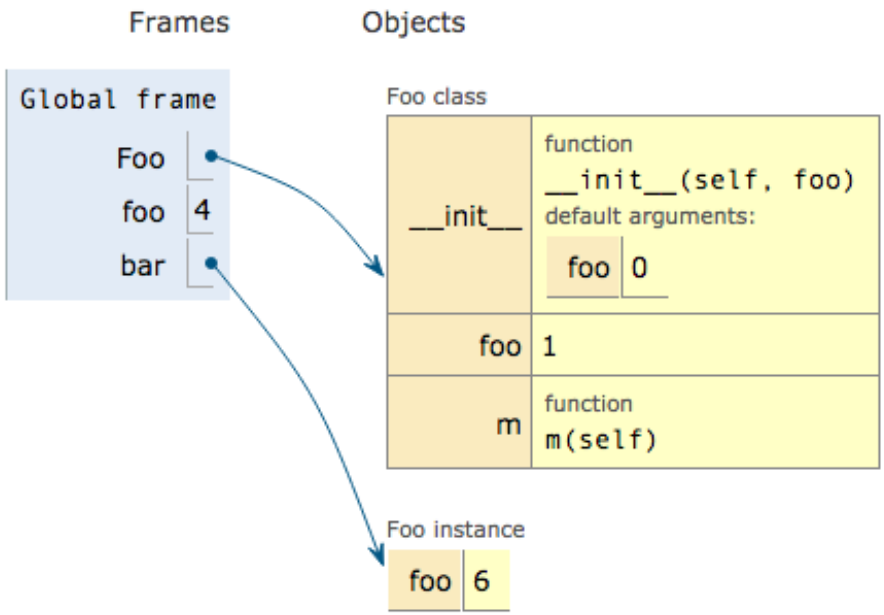


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self):
        return self.foo

foo = 4
bar = Foo(foo+2)
print(bar.m())
```

6

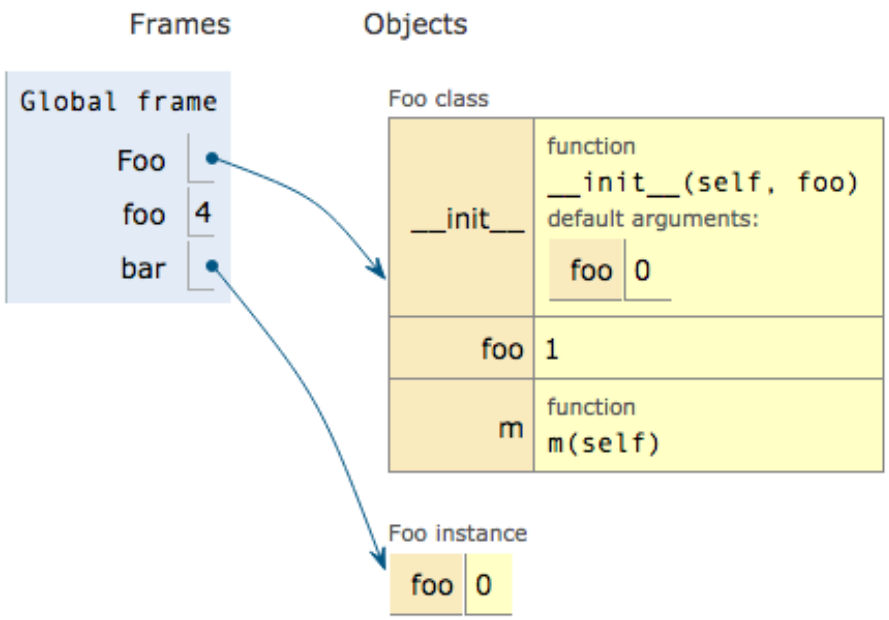


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self):
        return self.foo
```

```
foo = 4
bar = Foo()
print(bar.m())
```

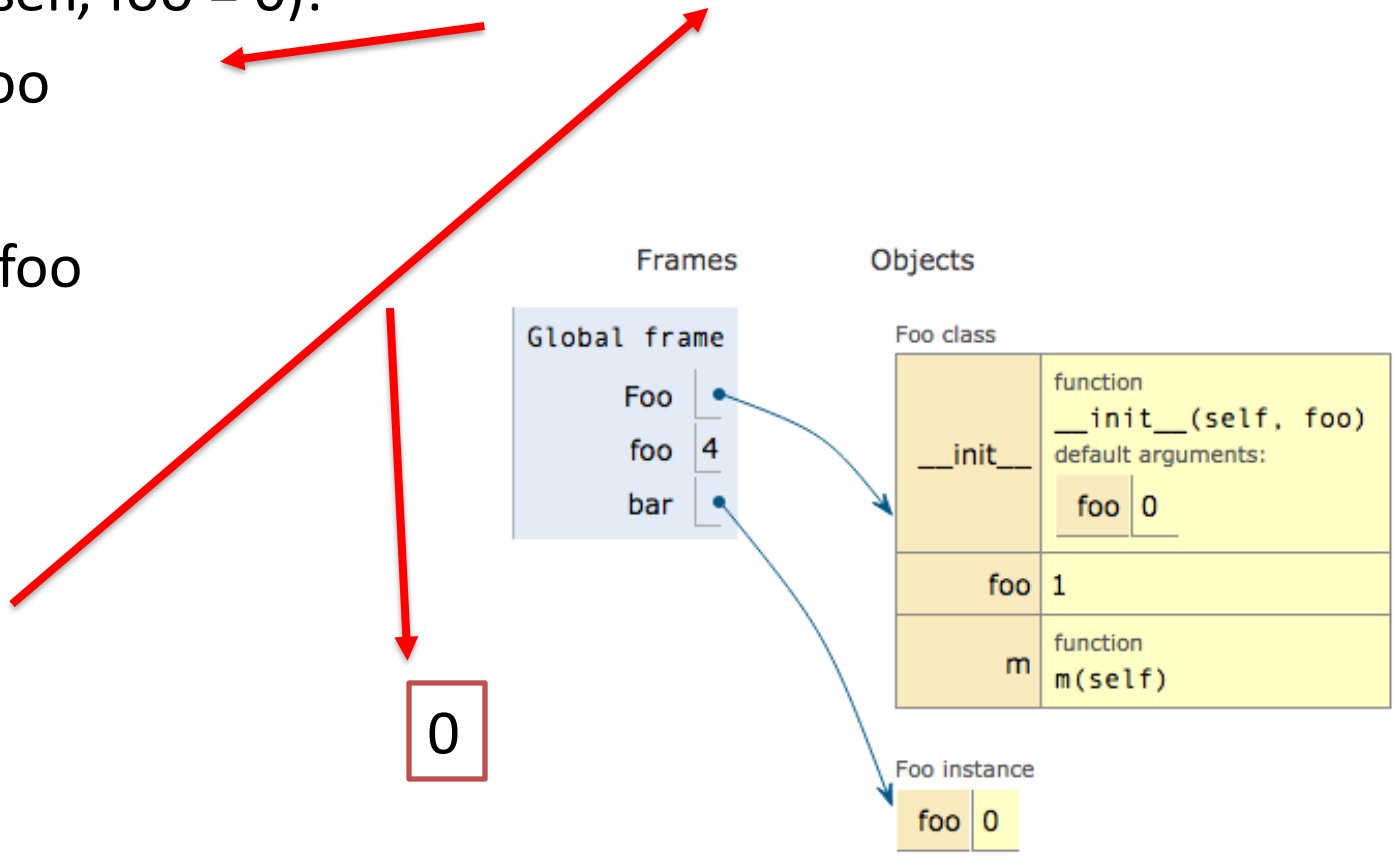
?



La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self):
        return self.foo

foo = 4
bar = Foo()
print(bar.m())
```

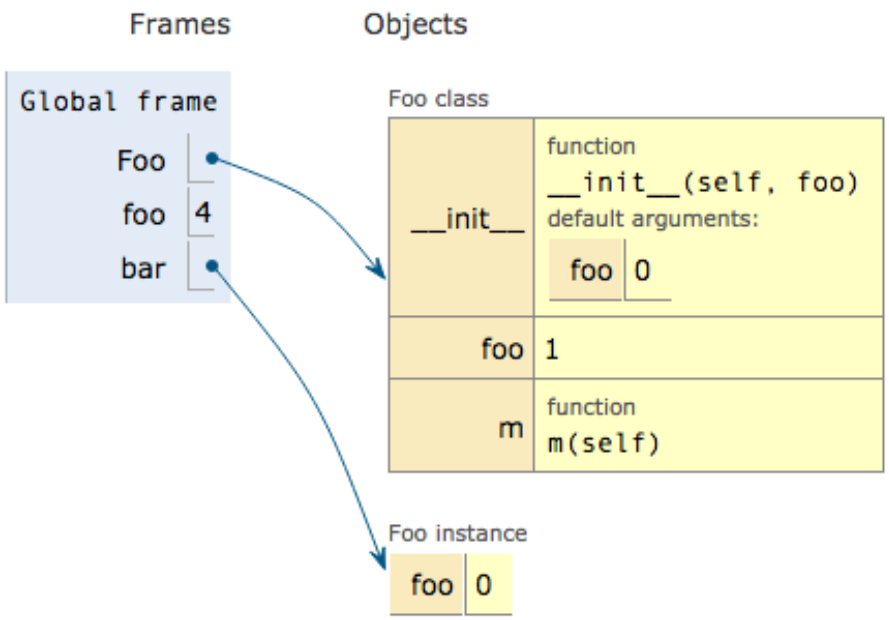


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self):
        return Foo.foo


foo = 4
bar = Foo()
print(bar.m())
```

?

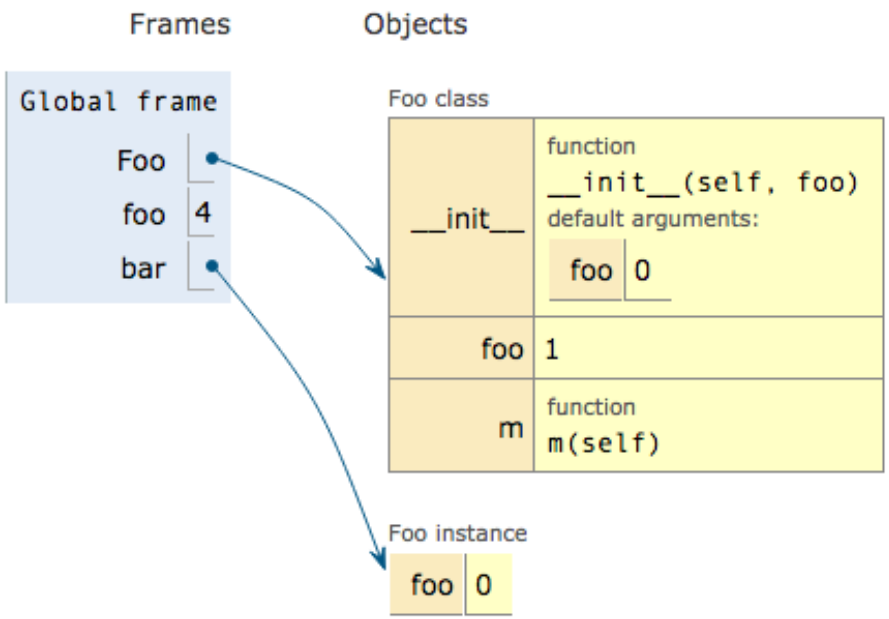


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        self.foo = foo
    def m(self):
        return Foo.foo

foo = 4
bar = Foo()
print(bar.m())
```

1

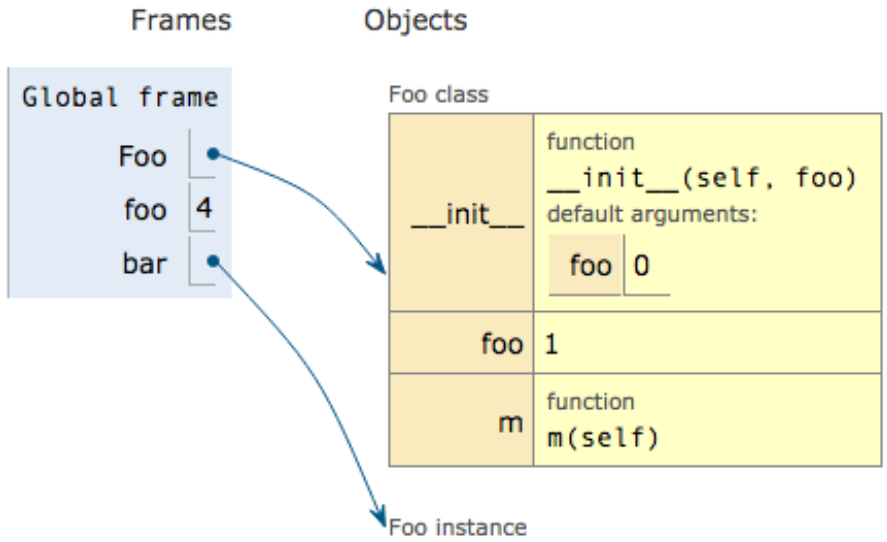


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        #self.foo = foo
        pass
    def m(self):
        return self.foo

foo = 4
bar = Foo()
print(bar.m())
```

?

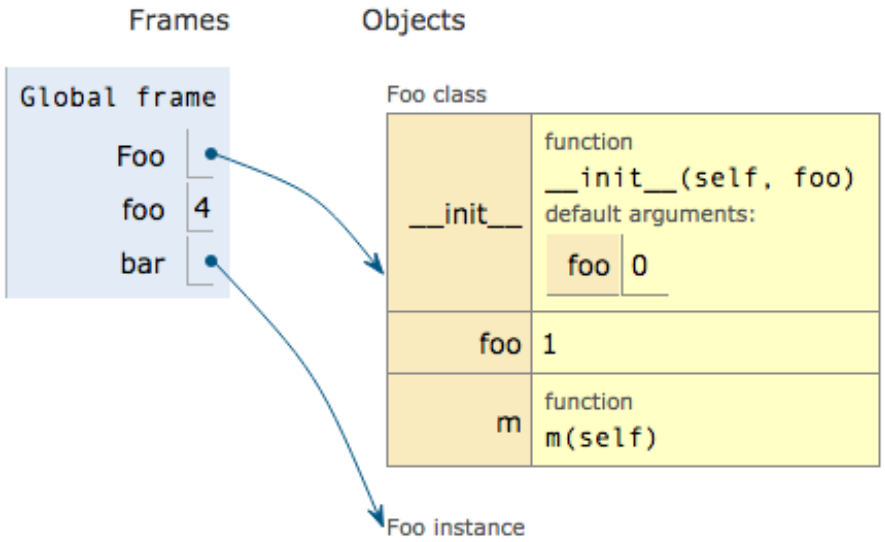


La portée des variables

```
class Foo :
    foo = 1
    def __init__(self, foo = 0):
        pass
    def m(self) :
        return self.foo

foo = 4
bar = Foo()
print(bar.m())
```

1



La portée des variables

```
class Etudiant :

    def __init__(self,n) :
        self.nom = n

    def nom(self) :
        return self.nom

kim = Etudiant("kim")
print(kim.nom())
```