

Python `zope.interface`

Stéphane Vialette

LIGM, Université Paris-Est Marne-la-Vallée

October 17, 2009

1 Introduction

Outline

1 Introduction

What are interfaces?

Description

Interfaces are objects that specify (document) the external behavior of objects that "provide" them. An interface specifies behavior through:

- Informal documentation in a docstring,
- Attribute definitions,
- Invariants, which are conditions that must hold for objects that provide the interface.

Interfaces provide a number of important benefits:

- Documentation,
- Specification,
- Support for component systems.

zope.interface: Defining interfaces

Example

```
import zope.interface
```

```
class IFoo(zope.interface.Interface):
```

```
    """IFoo interface docstring ..."""
```

```
    att = zope.interface.Attribute("""att docstring ...""")
```

```
    def method(arg1, arg2 = None):
```

```
        """method docstring ..."""
```

```
print type(IFoo)           # <class 'zope.interface.interface.InterfaceClass'>
```

```
print IFoo.__doc__         # IFoo interface docstring ...
```

```
print IFoo.__name__       # IFoo
```

```
print IFoo.__module__     # __main__
```

zope.interface: Defining interfaces

Description

The interface defined two attributes:

- `att`: This is the simplest form of attribute definition. It has a name and a docstring. It doesn't formally specify anything else.
- `method`: This is a method. A method is defined via a function definition. A method is simply an attribute constrained to be a callable with a particular signature, as provided by the function definition.

Note that `method` doesn't take a `self` argument.

zope.interface: Defining interfaces

Example

```
from ifoo import IFoo
```

```
att = IFoo['att']
```

```
print att # <zope.interface.interface.Attribute object at 0x957b72c>
```

```
print att.__doc__ # att docstring ...
```

```
print IFoo.get('att').__name__ # att
```

```
print IFoo.get('other_att') # None
```

```
print 'att' in IFoo # True
```

```
names = list(IFoo)
```

```
names.sort()
```

```
print names # ['att', 'method']
```

zope.interface: Defining interfaces

Example

```
from ifoo import IFoo
```

```
# interfaces are not classes
```

```
IFoo.att # raise AttributeError: 'InterfaceClass' object has no attribute 'att'
```

```
# Methods provide access to the method signature
```

```
method = IFoo['method']
```

```
print method.getSignatureString() # (arg1, arg2=None)
```

```
# But ...
```

```
att = IFoo['att']
```

```
print att.getSignatureString() # AttributeError: 'Attribute' object has no attribute 'getSignatureString'
```

zope.interface: Declaring interfaces

Description

- **provide:**

We say that objects provide interfaces. If an object provides an interface, then the interface specifies the behavior of the object. In other words, interfaces specify the behavior of the objects that provide them.

- **implement:** If a class implements an interface, then the instances of the class provide the interface. Objects provide interfaces that their classes implement. (Objects can provide interfaces directly, in addition to what their classes implement.)

It is important to note that classes don't usually provide the interfaces that they implement.

zope.interface: Declaring implemented interfaces

Description

There are several ways that you can assert that an object implements an interface:

- Call `zope.interface.implements` in your class definition.
- Call `zope.interfaces.directlyProvides` on your object.
- Call `zope.interface.classImplements` to assert that instances of a class implement an interface. For example

```
from zope.interface import classImplements  
classImplements(some_class, some_interface)
```

This approach is useful when it is not an option to modify the class source. Note that this doesn't affect what the class itself implements, but only what its instances implement.

zope.interface: Declaring implemented interfaces

Example

```
from ifoo import IFoo
import zope.interface

class Foo(object):
    # Foo implements IFoo
    zope.interface.implements(IFoo)

    def __init__(self, att = None):
        self.att = att

    def method(self, arg1, arg2 = None):
        return self.att, arg1, arg2

    def __repr__(self):
        return "Foo(%s)" % self.att
```

zope.interface: Declaring implemented interfaces

Example

```
# ask an interface whether it is implemented by a class  
print IFoo.implementedBy(Foo) # True
```

```
# ask whether an interface is provided by an object  
foo = Foo('value')  
print IFoo.providedBy(foo) # True
```

```
# Foo doesn't provide IFoo, it implements it  
print IFoo.providedBy(Foo) # False
```

```
# ask what interfaces are implemented by an object  
print list(zope.interface.implementedBy(Foo)) # [<InterfaceClass ifoo.IFoo>]
```

zope.interface: Declaring implemented interfaces

Example

```
#It is an error to ask for interfaces implemented by a non-callable object  
IFoo.implementedBy(foo)  
# TypeError: ('ImplementedBy called for non-factory', Foo(None))  
  
# ask what interfaces are provided by an object  
print list(zope.interface.providedBy(foo)) # [<InterfaceClass ifoo.IFoo>]  
print list(zope.interface.providedBy(Foo)) # []
```