Python assertIsInstance()

Summary: in this tutorial, you'll learn how to use Python `assertIsInstance()` method to test if an object is an instance of a class.

Introduction to the Python assertIsInstance() method

The `assertIsInstance()` is a method of the `TestCase` class of the unittest (https://www.pythontutorial.net/python-unit-testing/python-unittest/) module. The `assertIsInstance()` method tests if an object is an instance of a class (https://www.pythontutorial.net/python-oop/python-class/) .

The following shows the syntax of the `assertIsInstance()` method:

```
assertIsInstance(obj, cls, msg=None)
```

In this syntax:

- `obj` is the object to test.
- `cls` is a class or a tuple (https://www.pythontutorial.net/python-basics/python-tuples/) of classes.
- `msg` is an optional string that will be displayed if the `obj` is not an instance of the `cls` class.

Internally, the `assertIsInstance()` uses the `isinstance()` function to check if the object is an instance of the `cls` class.

If the `cls` is not the class of the obj but the base class of the class of the obj, the test will also pass.

Since the object class is the base class of all classes, the `assertIsInstance(obj, object)` will always pass.

Python assertIsInstance() method examples

Let's create a `shape.py` module with two classes `Shape` and `Square`. The `Shape` class is the base class of the `Square` class:

```
class Shape:
    pass


class Square(Shape):
    pass
```

To make it simple, the `Shape` and `Square` classes have no implementation.

1) Using the Python assertIsInstance() method example

The following example uses the `assertIsInstance()` method to test if the `square` object is an instance of the `Square` class:

```
import unittest

from shape import Shape, Square


class TestShape(unittest.TestCase):
    def setUp(self):
        self.square = Square()

    def test_is_instance(self):
        self.assertIsInstance(self.square, Square)
```

Run the test:

```
python -m unittest -v
```

Output:

```
test_is_instance (test_shape.TestShape) ... ok

----------------------------------------------------------------------
Ran 1 test in 0.001s

OK
```

Because the `square` instance variable is an object of the `Square` class, the test passes.

2) Using the Python assertIsInstance() method with a base class example

The following example uses the `assertIsInstance()` method to test if the `square` is an instance of the `Shape` class:

```
import unittest

from shape import Shape, Square


class TestShape(unittest.TestCase):
    def setUp(self):
        self.square = Square()

    def test_is_instance(self):
        self.assertIsInstance(self.square, Square)
    def test_is_instance_of_parent_class(self):
        self.assertIsInstance(self.square, Shape)
```

Run the test:

```
python -m unittest -v
```

Output:

```
test_is_instance (test_shape.TestShape) ... ok
test_is_instance_of_parent_class (test_shape.TestShape) ... ok

----------------------------------------------------------------------
Ran 2 tests in 0.001s

OK
```

3) Using the Python assertIsInstance() method to test if an object is an instance of the object class

The following example uses `assertIsInstance()` method to test if the `square` instance variable is an instance of the `object` class:

```
import unittest

from shape import Shape, Square


class TestShape(unittest.TestCase):
    def setUp(self):
        self.square = Square()

    def test_is_instance(self):
        self.assertIsInstance(self.square, Square)
    def test_is_instance_of_parent_class(self):
        self.assertIsInstance(self.square, Shape)

    def test_is_instance_of_object(self):
        self.assertIsInstance(self.square, object)
```

Run the test:

```
python -m unittest -v
```

Output:

```
test_is_instance (test_shape.TestShape) ... ok
test_is_instance_of_object (test_shape.TestShape) ... ok
test_is_instance_of_parent_class (test_shape.TestShape) ... ok

----------------------------------------------------------------------
Ran 3 tests in 0.001s

OK
```

Python assertIsNotInstance() method

The `assertIsNotInstance()` is the opposite of the `assertIsInstance()` method. It tests if an object is not an instance of a class:

```
assertNotIsInstance(obj, cls, msg=None)
```

For example:

```
import unittest

from shape import Shape, Square


```

```
class TestShape(unittest.TestCase):
    def setUp(self):
        self.square = Square()

    def test_is_instance(self):
        self.assertIsInstance(self.square, Square)

    def test_is_instance_of_parent_class(self):
        self.assertIsInstance(self.square, Shape)

    def test_is_instance_of_object(self):
        self.assertIsInstance(self.square, object)

    def test_is_not_instance(self):
        shape = Shape()
        self.assertNotIsInstance(shape, Square)
```

In this example, the `test_is_not_instance()` method uses the `assertIsNotInstance()` method to test if a `Shape` object is an instance of the `Square` class.

Run the test:

```
python -m unittest -v
```

Output:

```
test_is_instance (test_shape.TestShape) ... ok
test_is_instance_of_object (test_shape.TestShape) ... ok
test_is_instance_of_parent_class (test_shape.TestShape) ... ok
test_is_not_instance (test_shape.TestShape) ... ok

----------------------------------------------------------------------
Ran 4 tests in 0.002s

OK
```

Summary

- Use the `assertIsInstance()` method to test if an object is an instance of a class.
- Use the `assertIsNotInstance()` method to test if an object is not an instance of a class.