Python TUTORIAL

Python assertAlmostEqual()

Summary: in this tutorial, you learn how to use the Python `assertAlmostEqual()` method to test if two values are approximately equal.

Introduction to the Python assertAlmostEqual() method

The `assertAlmostEqual()` is a method of the `TestCase` class of the `unittest` module. The `assertAlmostEqual()` test if two values are approximately equal by doing the following:

- First, compute the difference.
- Second, round to the given number to decimal places (default 7)
- Third, compare the rounded value to zero.

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

```
assertAlmostEqual(first, second, places=7, msg=None, delta=None)
```

It uses the following check:

```
round(first-second, 7) == 0
```

The method uses places (or decimal places) to round the difference before comparing it to zero. Note that places are not significant digits.

If you pass a `delta` instead of `places` then the difference between first and second must be less or equal to (or greater than) delta.

The `assertAlmostEqual()` method allows you to use either places or delta. If you attempt to pass both arguments, you'll get a `TypeError` .

Python assertAlmostEqual method example

First, define a function `area()` that calculates the area of a circle in the `circle.py` file:

```
import math


def area(radius: float) -> float:
    return math.pi * math.pow(radius, 2)
```

The `area()` function accepts a radius as a float and returns the area of the circle as a float.

Since Python can only represent floats approximately (https://www.pythontutorial.net/advanced-python/python-float/) , you need to use the `assertAlmostEqual()` method to test the result of the `area()` with another float.

For example, the following test that uses the `assertEqual()` method will fail:

```
self.assertEqual(0.1+0.1+0.1, 0.3)
```

However, the following test that uses the `assertAlmostEqual()` method will pass:

```
self.assertAlmostEqual(0.1+0.1+0.1, 0.3)
```

Second, define a test module `test_circle.py` and import the `circle.py` module:

```
import unittest
from circle import area
from math import pi


class TestCircle(unittest.TestCase):
    def test_area(self):
        self.assertAlmostEqual(area(0), 0)
        self.assertAlmostEqual(area(1), pi)
        self.assertAlmostEqual(area(0.1), pi*0.1*0.1)
```

How it works:

- First, define a `TestCircle` class that inherits the `TestCase` class
- Second, add the `test_area()` test method to the `TestCircle` class.
- Third, use the `assertAlmostEqual()` method to test if the result of the `area()` function is almost equal to `0`, `pi`, and `pi * 0.1 * 0.1`.

Third, run the test:

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

Output:

```
test_area (test_circle.TestCircle) ... ok

----------------------------------------------------------------------
Ran 1 test in 0.000s

OK
```

Python assertNotAlmostEqual() method

The `assertNotAlmostEqual()` method is the opposite of the `assertAlmostEqual()` method. It tests if two values are not approximately equal.

Summary

- Use the Python `assertAlmostEqual()` method to test if two values are approximately equal.