Part 5: Using Unittest

Bookmark this page

Part 5: Using Unittest

Our first example showed the system that you would have to come up with if you wanted an automated way to assure that the Squarer class was working correctly.

Python also includes a built-in library for creating automated software tests: unittest. We can create tests in unittest that are very similar to the test_positive_numbers and test_negative_numbers tests that we created above, with unittest adding a few extra features. Let's create a test2.py file:

```
# test2.py
import unittest

from squarer import Squarer

class SquarerTest(unittest.TestCase):

    def test_positive_numbers(self):

        squares = {
            1: 1,
            2: 4,
            3: 9,
            12: 144,
            100: 10000,
        }

        for num, square in squares.items():
            self.assertEqual(square, Squarer.calc(num), "Squaring {}".format(num));

    def test_negative_numbers(self):

        squares = {
            -1: 1,
            -2: 4,
            -3: 9,
            -12: 144,
            -100: 10000,
        }

        for num, square in squares.items():
            self.assertEqual(square, Squarer.calc(num), "Squaring {}".format(num));
```

These test methods are just a little different than the test script we created for ourselves above. Here are the key differences:

```
import unittest                         # <---

from squarer import Squarer

class SquarerTest(unittest.TestCase): # <---
    ....
```

We import the unittest library, and our SquarerTest class inherits from unittest.TestCase. Next:

```
class SquarerTest(unittest.TestCase):
                                        # <---
    def test_positive_numbers(self):  # <---
```

Our test class methods *are not static*: they do not include a *@staticmethod* decorator, and they also accept *self* as their initial, implicit argument. Also, and this is very important, it is a *unittest requirement* that all of our test methods begin

initial, implicit argument. Also, and this is very important, it is a unittest requirement that all of our test methods begin with the word “test”, as in test_positive_numbers. Next:

‹ Previous Next ›

```
for num, square in squares.items():
    self.assertEqual(square, Squarer.calc(num), "Squaring {}".format(num));  # <---
```