Лабораторная работа №4

Автоматизированное тестирование
при помощи Python

Из чего состоит тест?

- **Test Case**

  Модуль тестирования. Проверка валидности ответа для определенного набора входов

- **Test Suite**

  Набор ТестКейсов. Содержит в себе тесты, которые должны быть запущены вместе

- **Test Fixture**

  Содержит в себе предварительные настройки для тестов (н-р, соединение с БД, создание временных директорий)

- **Test Runner/Loader**

  Модуль, запускающий тесты и выводящий результаты в нужном формате (может предоставлять графический/текстовый интерфейс)

Лабораторная работа №4

Основные методы TestCase

- setUp()
- tearDown()
- run(result=None)
- skipTest(reason)
- debug()
- fail(msg=None)
- countTestCases()
- id()
- doCleanups()

Основные методы TestCase

Method	Checks that	New in
assertEqual(a, b)	a == b	
assertNotEqual(a, b)	a != b	
assertTrue(x)	bool(x) is True	
assertFalse(x)	bool(x) is False	
assertIs(a, b)	a is b	2.7
assertIsNot(a, b)	a is not b	2.7
assertIsNone(x)	x is None	2.7
assertIsNotNone(x)	x is not None	2.7
assertIn(a, b)	a in b	2.7
assertNotIn(a, b)	a not in b	2.7
assertIsInstance(a, b)	isinstance(a, b)	2.7
assertNotIsInstance(a, b)	not isinstance(a, b)	2.7

Лабораторная работа №4

Основные методы TestCase

Method	Checks that	New in
assertAlmostEqual(a, b)	round(a-b, 7) == 0	
assertNotAlmostEqual(a, b)	round(a-b, 7) != 0	
assertGreater(a, b)	a > b	2.7
assertGreaterEqual(a, b)	a >= b	2.7
assertLess(a, b)	a < b	2.7
assertLessEqual(a, b)	a <= b	2.7
assertRegexpMatches(s, re)	regex.search(s)	2.7
assertNotRegexpMatches(s, re)	not regex.search(s)	2.7
assertItemsEqual(a, b)	sorted(a) == sorted(b) and works with unhashable objs	2.7
assertDictContainsSubset(a, b)	all the key/value pairs in *a* exist in *b*	2.7

Лабораторная работа №4

Основные методы TestCase

Method	Used to compare	New in
assertMultiLineEqual(a, b)	strings	2.7
assertSequenceEqual(a, b)	sequences	2.7
assertListEqual(a, b)	lists	2.7
assertTupleEqual(a, b)	tuples	2.7
assertSetEqual(a, b)	sets or frozensets	2.7
assertDictEqual(a, b)	dicts	2.7

Method	Checks that	New in
assertRaises(exc, fun, *args, **kwds)	fun(*args, **kwds) raises *exc*	
assertRaisesRegexp(exc, re, fun, *args, **kwds)	fun(*args, **kwds) raises *exc* and the message matches *re*	2.7

Лабораторная работа №4

Основные методы TestSuite

- addTest(test); test – TestCase/TestSuite
- addTests(tests)
- run(result)
- debug()
- countTestCases()
- __iter__()

Основные методы TestLoader

- loadTestsFromTestCase(testCase)
- loadTestsFromModule(module)
- getTestCaseNames(testCase)
- discover(*start_dir*, *pattern='test*.py'*, *top_level_dir=None*)
- sortTestMethodsUsing()
- suiteClass

Основные методы TestResult

- errors
- failures
- skipped
- expectedFailures
- unexpectedSuccesses
- shouldStop
- testsRun
- failfast
- stop()
- startTest(test)
- stopTest(test)
- startTestRun(test)
- stopTestRun(test)
- addFailure(test,err)
- addSuccess(test)
- addSkip(test,reason)
- addExpectedFailure(test,err)
- addUnexpectedSuccess(test)

Лабораторная работа №4

Пример. Создание теста

Имеется система классов:

```
class Animal:
        def speak(self):
                pass
class Cat(Animal):
        def __init__(self, name):
                self.name = name

        def speak(self):
                return "Meow"
```

```
class Dog(Animal):
        def __init__(self, name):
                self.name = name

        def speak(self):
                return "Wow"
```

Тестируем:

- Создание объекта
- Метод speak

Лабораторная работа №4

Пример. AnimalTestCase

```
from django.utils import unittest
from animal import *

class AnimalTestCase(unittest.TestCase):
    def setUp(self):
        self.cat = Cat('Tiger')
        self.dog = Dog('Bingo')

    def testSpeaking(self):
        self.assertEqual(self.cat.speak(), "Meow")
        self.assertEqual(self.dog.speak(), "Wow")

    def testName(self):
        self.assertEqual(self.cat.name, 'Tiger')
        self.assertEqual(self.dog.name, 'Bingo')

if __name__ == '__main__':
    unittest.main()
```

```
C:\Users\ivri\BitNami DjangoStack projects\sort>python animalTest.py
..
----------------------------------------------------------------------
Ran 2 tests in 0.003s

OK
```

Лабораторная работа №4

Пример. AnimalTestCase

Изменение степени выводимой информации:

Заменим

unittest.main()

На

```
suite = unittest.TestLoader().loadTestsFromTestCase(AnimalTestCase)
unittest.TextTestRunner(verbosity=2).run(suite)
```

Лабораторная работа №4

Пример. Вывод ошибок

Сделаем нарочно ошибку и посмотрим, что получится:

```
class Cat(Animal):
        def speak(self):
                return "Gav"
```

```
C:\Users\ivri\BitNami DjangoStack projects\sort>python animalTest.py
testName (__main__.AnimalTestCase) ... ok
testSpeaking (__main__.AnimalTestCase) ... FAIL

======================================================================
FAIL: testSpeaking (__main__.AnimalTestCase)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "animalTest.py", line 10, in testSpeaking
    self.assertEqual(self.cat.speak(), "Meow")
AssertionError: 'Gav' != 'Meow'

----------------------------------------------------------------------
Ran 2 tests in 0.036s

FAILED (failures=1)
```

Лабораторная работа №4

Пример. TestSuite

Итерация по тестам:

```
for test in suite:
        print test
```

Добавление тестов:

```
suite = unittest.TestSuite()
suite.addTest(AnimalTestCase('testSpeaking'))
suite.addTest(AnimalTestCase('testName'))
```

Пример. Пропуск тестов

```
@unittest.skip("demonstrating skipping")
  def testName(self): …
```

```
C:\Users\ivri\BitNami DjangoStack projects\sort>python animalTest.py
testName (__main__.AnimalTestCase) ... skipped 'demonstrating skipping'
testSpeaking (__main__.AnimalTestCase) ... ok

----------------------------------------------------------------------
Ran 2 tests in 0.009s

OK (skipped=1)
```

Помимо этого, используются:

- **@unittest.skipIf**(mylib.__version__ < (1, 3), "not supported in this library version")
- **@unittest.skipUnless**(sys.platform.startswith("win"), "requires Windows")
- Для класса: **@skip**("showing class skipping")

  class MySkippedTestCase(unittest.TestCase)

Лабораторная работа №4

Пример. Тест модели

Используем модель предыдущей лабораторной работы *books*

Файл *tests.py:*

```
from django.test import TestCase
class SimpleTest(TestCase):
    def test_basic_addition(self):
        """
        Tests that 1 + 1 always equals 2.
        """
        self.assertEqual(1 + 1, 2)
```

Лабораторная работа №4

Пример.Тест класса Publisher

```
class Publisher(models.Model):
        name = models.CharField(max_length=30)
        address = models.CharField(max_length=50)
        city = models.CharField(max_length=60)
        state_province = models.CharField(max_length=30)
        country = models.CharField(max_length=50)
        website = models.URLField()

        def __unicode__(self):
                return self.name
```

Тестируем: Создание объекта, строковое представление объекта, сохранение объекта, удаление объекта

Лабораторная работа №4

Пример.Тест класса Publisher

```
class PublisherTestCase(TestCase):
        def setUp(self):
                self.pub1=Publisher.objects.create(name="Drofa",
                address="Krasnopresnensyaya, 1",
                city="Moscow",state_province="Moscow",country="Russia",
                website="drofa.ru")

        def tearDown(self):
                self.pub1=None
```

Пример.Тест класса Publisher

```
    def testObjectAsString(self):
            self.assertEqual(str(self.pub1), 'Drofa')

    def testSaveObject(self):
            self.pub1.save()
            obj=Publisher.objects.get(name="Drofa")
            self.assertEqual(obj.name, 'Drofa')

    def testDeleteObject(self):
            self.pub1.save()
            Publisher.objects.filter(name="Drofa").delete()
self.assertNotEqual(Publisher.objects.filter(name="Drofa"),'Drofa')
```

Лабораторная работа №4

Пример.Тест класса Publisher

Тестирование шаблонов:

```
class TemplateTestCase(TestCase):
        def setUp(self):
                self.client=Client()

        def testBooksPage(self):
                response=self.client.get( "/books/" )
                self.assertContains( response, 'Author',status_code=200)
                self.assertTemplateUsed( response, 'books.html')

        def tearDown(self):
                self.client=None
```

Лабораторная работа №4