|  |  |  |
| --- | --- | --- |
| Защищено:    Шаповалов А.А.      "\_\_"\_\_\_\_\_\_\_\_\_\_\_2022 г. |  | Демонстрация:  Шаповалов А.А.      "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |

**Отчет по лабораторной работе № 5 по курсу**

**Базовые компоненты интернет-технологий**

## Тема работы: " Модульное тестирование в Python."

< # >

 (количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-52Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| Шаповалов Архип Аркадьевич | "28"\_\_января \_\_2022  г. |

Москва, МГТУ   -  2022

**1. Цель лабораторной работы:**

Изучение возможностей модульного тестирования в языке Python.

**2. Индивидуальное задание номер 1**

**Формулировка:**

1. Выберите любой фрагмент кода из лабораторных работ 1 или 2 или 3-4.
2. Модифицируйте код таким образом, чтобы он был пригоден для модульного тестирования.
3. Разработайте модульные тесты. В модульных тестах необходимо применить следующие технологии:
   * TDD - фреймворк (не менее 3 тестов).
   * BDD - фреймворк (не менее 3 тестов).

**Листинг программы:**

**Field.py**

from \_\_future\_\_ import annotations  
  
  
def field(items: tuple[dict] | list[dict], \*args: str):  
  
 if type(items) not in (list, tuple):  
 raise TypeError("Wrong goods format! Try correct types.")  
  
 if not (len(args) > 0):  
 raise RuntimeError('No keys specified to print!')  
  
 if tuple(arg for arg in args if arg in items[0].keys()) != args:  
 raise KeyError('Selected keys are not in goods list!')  
  
 if len(args) == 1:  
 for product in items:  
 yield product[args[0]]  
 else:  
 for product in items:  
 yield {arg: product[arg] for arg in args}  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 goods = [  
 {'title': 'Ковер', 'price': 2000, 'color': 'green'},  
 {'title': 'Диван для отдыха', 'price': 5300, 'color': 'black'}  
 ]  
  
 print("Testing...")  
  
 # Testing  
 assert tuple(i for i in field(goods, 'title')) == ('Ковер', 'Диван для отдыха'), \  
 "\n[TestError][#1] -> -Failed-"  
 assert tuple(i for i in field(goods, 'title', 'price')) == ({'title': 'Ковер', 'price': 2000},  
 {'title': 'Диван для отдыха', 'price': 5300}), \  
 "\n[TestError][#2] -> -Failed-"  
  
 print("All tests passed!\n")  
  
 print("Run with args ('title, color'):")  
 # Run  
 for callback in field(goods, 'title', "color"):  
 print(callback)

**Tests.py**

from field import field  
from json import loads as load\_jstr  
from behave import \*  
  
  
@given('a goods list')  
def given\_a\_list(context):  
 context.list = load\_jstr(context.text)  
  
  
@given("fields #1 {f1} and #2 {f2}")  
def given\_a\_fields(context, f1: {str}, f2: {str}):  
 context.F1 = f1  
 context.F2 = f2  
  
  
@given("fields #1 {f1}")  
def given\_a\_field(context, f1: {str}):  
 context.F1 = f1  
 context.F2 = None  
  
  
@when('we call function with that data we get result')  
def func\_call(context):  
 if context.F2 is not None:  
 context.data = tuple(i for i in field(context.list, context.F1, context.F2))  
 return  
 context.data = tuple(i for i in field(context.list, context.F1))  
  
  
@Then("we can assert this data with the tuple")  
def check\_data(context):  
 assert context.data == tuple(load\_jstr(context.text))

**Unittests.py**

import unittest  
from field import field  
  
goods = [  
 {'title': 'Ковер', 'price': 2000, 'color': 'green'},  
 {'title': 'Диван для отдыха', 'price': 5300, 'color': 'black'}  
 ]  
  
class TestEquation(unittest.TestCase):  
  
  
 def test\_calculate(self):  
 self.assertEqual(tuple(i for i in field(goods, 'title')), ('Ковер', 'Диван для отдыха'))  
 self.assertEqual(tuple(i for i in field(goods, 'title', 'price')), ({'title': 'Ковер', 'price': 2000},  
 {'title': 'Диван для отдыха',  
 'price': 5300}))  
  
 def test\_type(self):  
 with self.assertRaises(TypeError) as e:  
 tuple(i for i in field('', 'title'))  
  
 def test\_runtime(self):  
 with self.assertRaises(RuntimeError) as e:  
 tuple(i for i in field(goods))  
  
 def test\_key(self):  
 with (self.assertRaises(KeyError)) as e:  
 tuple(i for i in field(goods, 'title1', 'price'))  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 unittest.main()

**Результат работы программы:**

**Field.py**

Testing...

All tests passed!

Run with args ('title, color'):

{'title': 'Ковер', 'color': 'green'}

{'title': 'Диван для отдыха', 'color': 'black'}

**Unittests.py**

Testing started at 15:04 ...

Ran 4 tests in 0.001s

OK

Launching unittests with arguments python -m unittest /Users/arhipshapovalov/PycharmProjects/LR5/unittests.py in /Users/arhipshapovalov/PycharmProjects/LR5