



**Московский государственный технический
университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчет по домашнему заданию

Выполнил:

студент группы ИУ5-31Б

Фонин Максим Алексеевич

Подпись и дата:

Проверил:

преподаватель каф. ИУ5

Гапанюк Ю.Е.

Подпись и дата:

Москва, 2021 г.

Описание задания:

1. Модифицируйте код лабораторной работы №6 таким образом, чтобы он был пригоден для модульного тестирования.
2. Используя материалы лабораторной работы №4 создайте модульные тесты с применением TDD - фреймворка (2 теста) и BDD - фреймворка (2 теста).

Текст программы:

Файл olejka_bot.py (это главный файл бота, остальные дополнительные файлы лежат на GitHub, а также указаны в отчете к 6 лабораторной работе)

```
from aiogram import Bot, types
from aiogram.utils import executor
from aiogram.dispatcher import Dispatcher
from aiogram.dispatcher.storage import FSMContext
from aiogram.contrib.fsm_storage.memory import MemoryStorage
from aiogram.contrib.middlewares.logging import LoggingMiddleware

import sys, os
sys.path.append(os.getcwd())

from config import TOKEN
from utils import TestStates
from messages import MESSAGES
from answers import *

bot = Bot(token=TOKEN)
dp = Dispatcher(bot, storage=MemoryStorage())

dp.middleware.setup(LoggingMiddleware())

@dp.message_handler(commands=['start'])
async def process_start_command(msg: types.Message):
    await msg.reply(MESSAGES['start'])

@dp.message_handler(commands=['help'])
async def process_help_command(msg: types.Message):
    await msg.reply(MESSAGES['help'])

@dp.message_handler(state='*', commands=['setstate'])
async def process_setstate_command(msg: types.Message, state: FSMContext):
    arg = msg.get_args()
    curr_state = dp.current_state(user=msg.from_user.id)
    if not arg:
        await curr_state.reset_state()
        return await msg.reply(MESSAGES['state_reset'])

    if (not arg.isdigit()) or (not int(arg) < len(TestStates.all())):
        return await msg.reply(MESSAGES['invalid_key'].format(key=arg))

    if int(arg) != 0:
        await curr_state.set_state(TestStates.all()[int(arg)])
```

```

        return await msg.reply(MESSAGES[int(arg)])
    else:
        await curr_state.set_state(TestStates.all()[int(arg)])
        user_data = await state.get_data()
        if (user_data.get("chosen_location" == None)) or
        (user_data.get("chosen_backpack" == None) or (user_data.get("chosen_transport"
        == None)):
            return await msg.reply("Заполните все данные", reply=False)
        await
        msg.reply(MESSAGES['camp'].format(loc=location[user_data.get("chosen_location")],
        bp=backpack[user_data.get("chosen_backpack")],
        transp=transport[user_data.get("chosen_transport")]))

@dp.message_handler(state=TestStates.TEST_STATE_1)
async def first_test_state_choice(msg: types.Message, state: FSMContext):
    answer = msg.text
    if (not answer.isdigit()) or (not int(answer) < len(location)):
        return await msg.reply(MESSAGES['wrong_answer'], reply=False)

    await state.update_data(chosen_location=int(answer))
    await msg.reply(MESSAGES['success_answer'], reply=False)

@dp.message_handler(state=TestStates.TEST_STATE_2)
async def second_test_state_case_met(msg: types.Message, state: FSMContext):
    answer = msg.text
    if (not answer.isdigit()) or (not int(answer) < len(backpack)):
        return await msg.reply(MESSAGES['wrong_answer'], reply=False)

    await state.update_data(chosen_backpack=int(answer))
    await msg.reply(MESSAGES['success_answer'], reply=False)

@dp.message_handler(state=TestStates.TEST_STATE_3)
async def third_test_state_case_met(msg: types.Message, state: FSMContext):
    answer = msg.text
    if (not answer.isdigit()) or (not int(answer) < len(transport)):
        return await msg.reply(MESSAGES['wrong_answer'], reply=False)

    await state.update_data(chosen_transport=int(answer))
    await msg.reply(MESSAGES['success_answer'], reply=False)

async def shutdown(dispatcher: Dispatcher):
    await dispatcher.storage.close()
    await dispatcher.storage.wait_closed()

if __name__ == '__main__':
    executor.start_polling(dp, on_shutdown=shutdown)

```

Файл TDD_tests.py:

```

import unittest
import sys, os

sys.path.append(os.getcwd())

```

```
from bot.messages import first_state_message, second_state_message
```

```
class TestMessages(unittest.TestCase):
    def test_first_state_message(self):
        res = 'Определимся, куда нам собираться:\n' \
              '0. Лес\n' \
              '1. Горы\n' \
              '2. Пустыня'
        self.assertEqual(first_state_message, res)

    def test_secon_state_message(self):
        res = 'Что возьмем с собой?\n' \
              '0. Палатка\n' \
              '1. Карта\n' \
              '2. Фен'
        self.assertEqual(second_state_message, res)

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

Файл BDD_tests.feature:

Feature: Testing camp **set**

Scenario: test setting camp **set with** answers 2 0 1

Given answers **for** states 1, 2, 3 : 2, 0, 1

When setting camp

Then we should see В итоге: Локация: Пустыня. В рюкзаке: Палатка. Способ передвижения: Вертолет. /setstate, чтобы очистить и собрать заново.

Scenario: test setting camp **set with** answers 1 2 2

Given answers **for** states 1, 2, 3 : 1, 2, 2

When setting camp

Then we should see В итоге: Локация: Горы. В рюкзаке: Фен. Способ передвижения: Караван верблюдов. /setstate, чтобы очистить и собрать заново.

Файл steps.py:

```
from behave import Given, When, Then
import os, sys
```

```
sys.path.append(os.getcwd())
from bot.messages import camp_message
from bot.answers import location, backpack, transport
```

```
@Given("answers for states 1, 2, 3 : {num1}, {num2}, {num3}")
```

```
def given_answers(context, num1, num2, num3):
```

```
    context.ans1 = int(num1)
```

```
    context.ans2 = int(num2)
```

```
    context.ans3 = int(num3)
```

```
@When("setting camp")
```

```
def setting_camp(context):
```

```
    res = camp_message.format(loc=location.get(context.ans1),
```

```
bp=backpack.get(context.ans2), transp=transport.get(context.ans3))
```

```
    context.result = res
```

```

@Then("we should see В итоге: Локация: {loc}. В рюкзаке: {bp}. Способ
передвижения: {transp}. /setstate, чтобы очистить и собрать заново.")
def check_result(context, loc, bp, transp):
    res = 'В итоге:\nЛокация: ' + str(loc) + '.\nВ рюкзаке: ' + str(bp) +
    '.\nСпособ передвижения: ' + str(transp) + '.\n/setstate, чтобы очистить и
собрать заново.'
    assert(context.result == res)

```

Примеры выполнения программы:

TDD:

```

PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
(aiogram-venv) f0max@f0max:~/Downloads/GitHub/BKIT_2021/DZ$ cd /home/f0max/Downloads/GitHub/BKIT_2021/DZ ; /usr/bin/env /home/f0max/Downloads/GitHub/BKIT_2021/DZ/aiogram-venv/bin/python /
home/f0max/.vscode/extensions/ms-python.python-2021.10.1365161279/pythonFiles/lib/python/debugpy/launcher 44183 -- /home/f0max/Downloads/GitHub/BKIT_2021/DZ/TDD/TDD_tests.py
..
-----
Ran 2 tests in 0.000s
OK
(aiogram-venv) f0max@f0max:~/Downloads/GitHub/BKIT_2021/DZ$ █

```

BDD:

```

PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
(aiogram-venv) f0max@f0max:~/Downloads/GitHub/BKIT_2021/DZ$ behave
Feature: Testing camp set # features/BDD_tests.feature:1

  Scenario: test setting camp set with answers 2 0 1 # features/BDD_tests.feature:2
    Given answers for states 1, 2, 3 : 2, 0, 1 # features/steps/steps.py:9 0.000s
    When setting camp # features/steps/steps.py:15 0.000s
    Then we should see В итоге: Локация: Пустыня. В рюкзаке: Палатка. Способ передвижения: Вертолет. /setstate, чтобы очистить и собрать заново. # features/steps/steps.py:20 0.000s

  Scenario: test setting camp set with answers 1 2 2 # features/BDD_tests.feature:7
    Given answers for states 1, 2, 3 : 1, 2, 2 # features/steps/steps.py:9 0.000s
    When setting camp # features/steps/steps.py:15 0.000s
    Then we should see В итоге: Локация: Горы. В рюкзаке: Фен. Способ передвижения: Караван верблюдов. /setstate, чтобы очистить и собрать заново. # features/steps/steps.py:20 0.000s

1 feature passed, 0 failed, 0 skipped
2 scenarios passed, 0 failed, 0 skipped
6 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.002s
(aiogram-venv) f0max@f0max:~/Downloads/GitHub/BKIT_2021/DZ$ █

```