Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №3

Технології розроблення програмного забезпечення

ШАБЛОНИ «SINGLETON», «ITERATOR», «PROXY», «STATE», «STRATEGY»

Варіант 24

Виконала:
студентка групи ІА-14
Шеліхова А.О

Перевірив:
Мягкий М.Ю.

Київ 2023

Тема: ШАБЛОНИ «SINGLETON», «ITERATOR», «PROXY», «STATE», «STRATEGY»

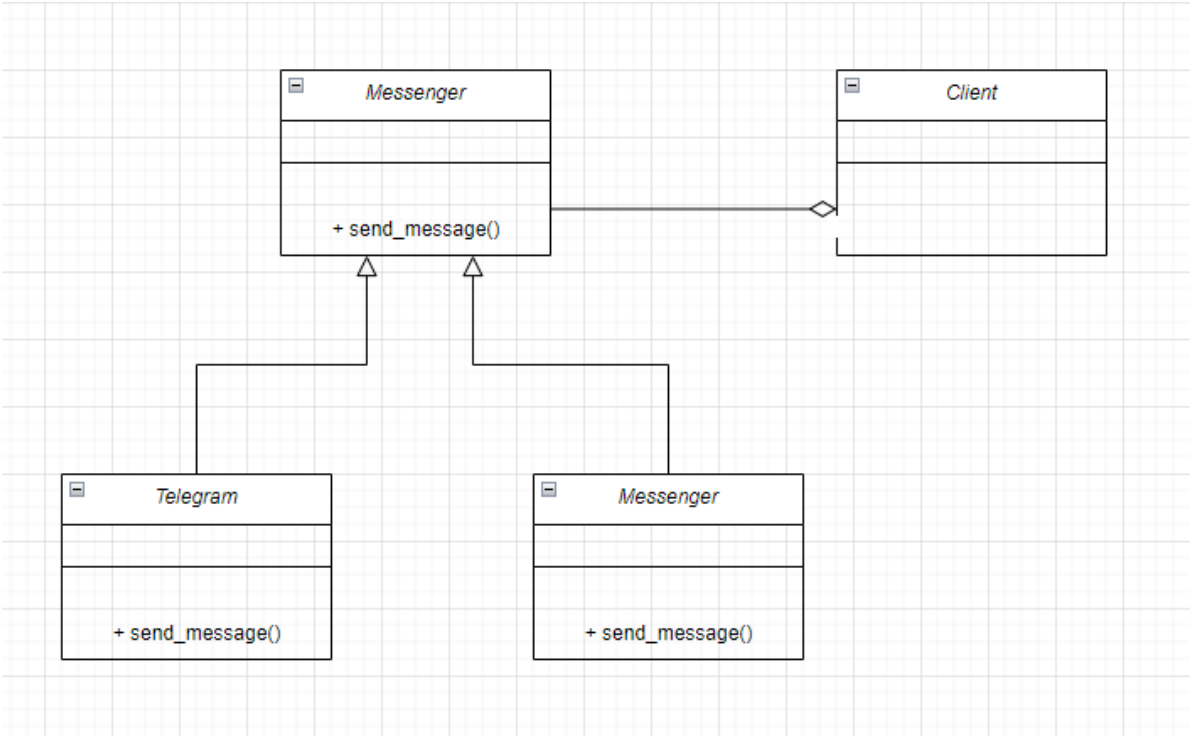
Інструмент автоматизації повинен забезпечувати найпростіші автоматичні дії для зручності користувача: завантаження нових фільмів / книг / файлів при випуску (наприклад, щоп'ятниці з'являються нові серії улюблених серіалів); встановити статуси в комунікаторах (skype - away при нульовій активності на тривалий час) і т.д. Автоматизація забезпечується шляхом введення правил (на зразок IFTTT.com сервісу), запису макросів (натискання клавіш, дії миші), планувальника завдань (о 5 ранку - початок роздачі торрент-файлів).

Порядок виконання роботи

1. Ознайомитися з короткими теоретичними відомостями.
2. Реалізувати частину функціоналу робочої програми у вигляді класів та їхньої взаємодії для досягнення конкретних функціональних можливостей.
3. Застосування одного з розглянутих шаблонів при реалізації програми.

У ході роботи була реалізована програма для регулярної відправки повідомлень про погоду у вибраному місті у вибраний мессенджер. Програма пропонує користувачеві ввести місто, в якому буде відслідковуватися погода, вибрати мессенджер, куди відправляти повідомлення (Telegram або Viber) та вибрати одиниці вимірювання (standard, metric або imperial). Після чого користувач кожні 20 секунд отримує повідомлення про погоду. Вибір мессенджера реалізовано за допомогою шаблону “Strategy”.

Шаблон “Strategy”:



Код програми:

Класи, які реалізують функціонал мессенджерів:

```
from abc import ABC, abstractmethod

import requests as reqs


class Messenger(ABC):

    @abstractmethod
    def send_message(self, bot_token, receiver, message_type, message_content):
        pass


class Telegram(Messenger):

    def send_message(self, bot_token, receiver, message):

        url = f"https://api.telegram.org/bot{bot_token}/sendMessage"

        data = {
            'chat_id': receiver,
            'parse_mode': "Markdown",
            'text': message
        }

        success = reqs.post(url, data=data).json()['ok']

        return success


class Viber(Messenger):

    def send_message(self, bot_token, receiver, message):

        url = "https://chatapi.viber.com/pa/send_message"

        headers = {'X-Viber-Auth-Token': bot_token}

        data = {
            "receiver": receiver,
            "type": "text",
            "text": message
        }

        success = reqs.post(url, headers=headers, json=data).json()['status_message'] == 'ok'

        return success


MESSENGERS = {
    'telegram': Telegram(),
    'viber': Viber()
}
```

Головний код програми:

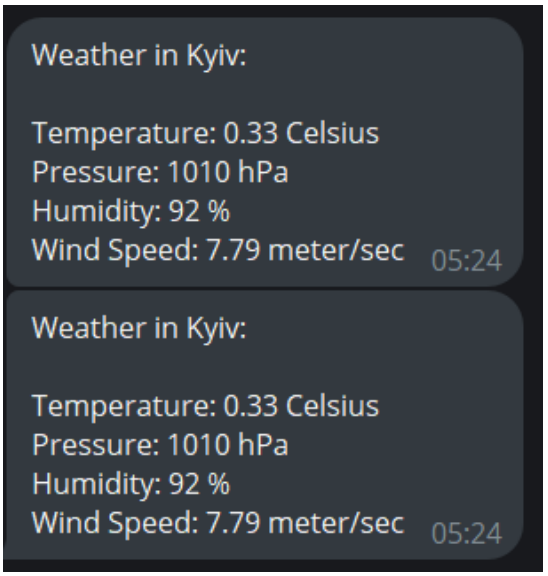
```
import os

from datetime import datetime as dt

from apscheduler.schedulers.background import BackgroundScheduler
```

```
from messengers import MESSENGERS
from utils import get_weather
from config import API_KEYS, OPENWEATHERAPP_API_KEY, WEATHER_UNITS, SEND_MESSAGE_EVERY


def execute(function):

    def executor():

        success = function()

        print(f"[{dt.now().strftime('%d-%m-%Y %H:%M:%S')}] {'Success' if success else 'Failed'}.")

    return executor


def main():

    success = False

    weather_units = list(WEATHER_UNITS.keys())

    messengers = list(MESSENGERS.keys())

    while not success:

        try:

            city = input("Enter city to get weather in: ")

            print("\nChoose unit:\n\t" + '\n\t'.join([str(i + 1) + ': ' + weather_units[i] for i in
range(len(weather_units))]))

            weather_unit_index = int(input("Unit: ")) - 1

            try:
                weather_unit = weather_units[weather_unit_index]
            except IndexError:
                print("\nInvalid unit\n")
                continue

            print("\nChoose messenger:\n\t" + '\n\t'.join([str(i + 1) + ': ' + messengers[i] for i in
range(len(messengers))]))

            messenger_index = int(input("Messenger: ")) - 1

            try:
                messenger = messengers[messenger_index]
            except IndexError:
                print("\nInvalid messenger\n")
                continue

            receiver = input("Enter receiver: ")

            weather_result = get_weather(OPENWEATHERAPP_API_KEY, city, weather_unit)

            if not weather_result['success']:
                print(f"\n{weather_result['error']}\n")
                continue

            weather = weather_result['weather']

            message = f"Weather in {city.title()}:\n\n" \
                f"Temperature: {weather['temperature']} {WEATHER_UNITS[weather_unit]['temperature']}\n" \
                f"Pressure: {weather['pressure']} hPa\n" \
                f"Humidity: {weather['humidity']} %\n" \
                f"Wind Speed: {weather['wind_speed']} {WEATHER_UNITS[weather_unit]['wind_speed']}"

            scheduler = BackgroundScheduler()

            scheduler.add_job(
                func=execute(lambda: MESSENGERS[messenger].send_message(API_KEYS[messenger], receiver,
message)),
                trigger="interval",
```

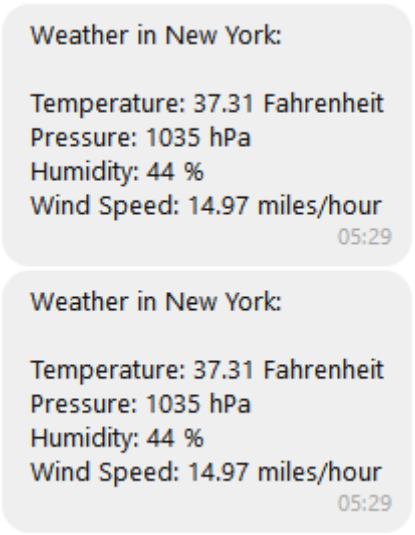
```
                seconds=SEND_MESSAGE_EVERY
            )

            scheduler.start()

            success = True

        except Exception as exception:
            print(f"\n{type(exception).__name__}: {exception}\n")

    print("\nSuccess!\n")

    while True:
        pass


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

Результати:

Погода в Києві у одиницях вимірювання metric (Telegram):



Погода в Нью Йорку у одиницях вимірювання imperial (Viber):



Висновок: під час виконання лабораторної роботи було проведено ознайомлення з теоретичними відомостями та реалізувано частину функціоналу робочої програми у вигляді класів та їхньої взаємодії для досягнення конкретних функціональних можливостей із застосуванням одного з розглянутих шаблонів при реалізації програми.