Московский Государственный Технический Университет им. Н.Э. Баумана

Разработка интернет-приложений

Отчёт по лабораторной работе №3

«Python классы»

Выполнил:

студент группы ИУ5-51

Никитин К.И.

1. Цель работы

В лабораторной работе необходимо создать набор классов для реализации работы с VK API.

2. Листинг программы

Модуль `derived_client.py`

```
import requests

class BaseClient:
    # URL vk api
    BASE_URL = None
    # метод vk api
    method = None
    # GET, POST, ...
    http_method = None


    # Получение GET параметров запроса
    def get_params(self):
        return None


    # Получение данных POST запроса
    def get_json(self):
        return None


    # Получение HTTP заголовков
    def get_headers(self):
        return None

    # Склейка url
    def generate_url(self, method):
        return '{0}{1}'.format(self.BASE_URL, method)

    # Отправка запроса к VK API
    def _get_data(self, method, http_method):
        resp = requests.get(self.BASE_URL + self.method,
params=self.get_params())
        return self.response_handler(resp)

    # Обработка ответа от VK API
    def response_handler(self, response):
        return response

    # Запуск клиента
    def execute(self):
        return self._get_data(
            self.method,
            http_method=self.http_method
        )
```

Модуль get_user_id.py

```
import json
import base_client


class GetUserId(base_client.BaseClient):
    BASE_URL = "https://api.vk.com/method/"
    method = "users.get"
    http_method = "GET"
    user_ids = ""

    def get_params(self):
        return {
            "user_ids": self.user_ids
        }

    def get_json(self, data):
        return json.dumps(data)

    def response_handler(self, response):
        resp = response.json()
        return resp
```

Модуль derived_client.py

```
import base_client
import json
import get_user_id
import sys
import matplotlib.pyplot as plt
import numpy as np
from _datetime import datetime

def calculate_age(born):
    today = datetime.utcnow()
    return today.year - born.year -((today.month, today.day) < (born.month,
born.day))

def draw_distribution(array, draw_hist):
    #find min and max in ages to determine range of distribution
    min = array[0]
    max = array[0]
    for i in range(1, len(array)):
        if array[i] < min:
            min = array[i]
        if array[i] > max:
            max = array[i]
    #initialize and count number of people in each age
    distribution = {}
    for i in range(min, max + 1):
        distribution[i] = 0
    for i in range(0, len(array)):
        #key in dict is age
        distribution[array[i]] += 1
    #find max count in distribution
    dmax = 0
    for i in distribution:
        if distribution[i] > dmax:
            dmax = distribution[i]
    #x/dmax, where x is length of the highest column
    relation = 80/dmax;
```

```
    #normalize distribution by relation
    x = []
    y = []
    for i in distribution:
        distribution[i] *= relation
        distribution[i] = int(round(distribution[i], 0))
        x.append(i)
        y.append(distribution[i])
    if(draw_hist):
        plt.hist([array], range(min, max + 1))
        plt.show()
    else:
        #print distribution
        for i in distribution:
            sys.stdout.write(str(i))
            sys.stdout.write(" ")
            for i in range(0, distribution[i]):
                sys.stdout.write("#")
            print('');

class GetFriends(base_client.BaseClient):
    BASE_URL = "https://api.vk.com/method/"
    method = "friends.get"
    http_method = "GET"
    user_id = ""

    def get_params(self):
        return {
            "user_id": self.user_id,
            "fields": "bdate"
        }

    def get_json(self, data):
        return json.dumps(data)

    def response_handler(self, response):
        resp = json.loads(response.text)
        return resp

g = get_user_id.GetUserId()
g.user_ids = "durov"
user = g.execute()
print(user)

uid = user["response"][0]["uid"]
print(uid)

c = GetFriends()
c.user_id = uid
friends = c.execute()
print(friends)

ages = []

for f in friends["response"]:
    if "bdate" in f:
        if len(f["bdate"]) > 5:
            ages.append(calculate_age(datetime.strptime(f["bdate"],
"%d.%m.%Y")))

print(ages)
draw_distribution(ages, False)

plt.show()
```

3. Результат работы

Никнейм пользователя: oblomoff (3000+ друзей)
В режиме вывода в консоль:

```
13 #
14 ######
15 ###########
16 ####################
17 ####################
18 ########################################
19 ############################################################
20 ########################################
21 ##############################################################
22 ############################################################
23 ##################################################################
24 ##########################################################
25 ########################################################################
26 ############################################################
27 ##########################################################################
28 ##################################################################################
29 ##############################################################
30 ##########################################################
31 ##############################
32 ####################################
33 ###################################
34 ########################
35 ##################
36 #########
37 ##################
38 ########
39 ########
40 #######
41 ####
42 ###
43 ###
```

В режиме построения графика: