Отчет по лабораторной работе № 3 по курсу

«РИП»

Тема работы: "Python классы**"**

6

(количество листов)

Москва, МГТУ – 2017

Оглавление

Задание

Вход:
username или vk_id пользователя
Выход:
Гистограмма распределения возрастов друзей пользователя, поступившего на вход
Пример: Вход: reigning **Выход:**
19 #
20 ##
21 ##
22 ####################
23 ################
24 ####
25 #
28 #
29 #
30 #
37 #
38 ##
45 #

Выполнение

За основу возьмем базовый класс:
https://gist.github.com/Abashinos/024c1dcaf92f1ff733c63a07e447ab51

Для реализации методов ВК будем наследоваться от этого базового класса. В классе наследнике реализуем методы:
● get_params если
есть get параметры (необязательно)
● get_json если
нужно передать данные (необязательно)
● get_headers если
нужно передать дополнительные заголовки (необязательно)
● response_handler обработчик
ответа. В случае успешного ответа необходим,
чтобы преобразовать результат. В случае ошибочного ответа необходим, чтобы сформировать исключение

Для решения задачи обратимся к двум метод VK API
1) users.get для получения vk id по username
2) friends.get для получения друзей пользователя. В этом методе нужно передать в get параметрах fields=bdate для получения возраста. Так же создадим исключение для игнорирования тех, у кого не указана дата рождения.

base.py

В этом файле опишем базовый класс, при этом укажем параметры выполнения запроса.

```
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):
        response = None

        # todo выполнить запрос

        return self.response_handler(response)

    # 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
        )
```

uid.py

В этом файле наследуем базовый класс и используем метод users.get.

```
from base import *
import requests
import json
class GetID(BaseClient):
    BASE_URL = 'https://api.vk.com/method/users.get'
    http_method = 'GET'
    def __init__(self, name):
```

```
        self.name = name
    def get_params(self):
        return 'user_ids=' + self.name
    def response_handler(self, response):
        try:
            obje = json.loads(response.text)
            return obje.get('response')[0].get('uid')
        except:
            raise Exception("Данный пользователь не найден
{}".format(self.name))

    def _get_data(self, method, http_method):
        response = None
        response = requests.get(self.BASE_URL + '?' + self.get_params())
        return self.response_handler(response)
```

friends.py

В этом файле так происходит наследование, но уже используем метод friends.get для получения списка друзей.

```
from base import *
import requests
import json
from datetime import datetime
class GetFriends(BaseClient):
    BASE_URL = 'https://api.vk.com/method/friends.get'
    http_method = 'GET'
    def __init__(self,uid):
        self.uid = uid
    def  get_params(self):
        return 'user_id=' + str(self.uid) + '&fields=bdate'
    def response_handler(self, response):
        try:
            obje = json.loads(response.text)
            friends = obje.get('response')
            ages = []
            for friend in friends:
              b_date = friend.get('bdate')
              if b_date is None or b_date.count('.') < 2:
                  continue
              b_date = datetime.strptime(b_date, "%d.%m.%Y")
              n_date = datetime.now()
              ages.append(int((n_date - b_date).days) // 365.2425)
            uniqages = list(set(ages))
            return sorted([(x, ages.count(x)) for x in uniqages], key=lambda
x: x[0])
        except:
            raise Exception("У пользователя нет друзей, либо они недоступны
{}".format(self.uid))

    def _get_data(self, method, http_method):
        response = requests.get(self.BASE_URL + '?' + self.get_params())
        return self.response_handler(response)
```

main.py

В этом файле выполняются запросы, а так же построение графиков, на основе полученных данных с помощью библиотеки *matplotlib,* тем самым выполним доп. задание.

```
import numpy as np
import matplotlib.mlab as mlab
import matplotlib.pyplot as plt
from uid import *
from friends import *


input_id = input('Введите id: ')
user = GetID(input_id)
user_id = user.execute()
friends_client = GetFriends(user_id)
friends = friends_client.execute()
ages = []
counts = []
for (age, count) in friends:
    print('{} {}'.format(int(age), int(count)))
    ages.append(int(age))
    counts.append(int(count))
plt.grid()
plt.minorticks_on()
plt.axis([0, 120, 0, 50])
plt.figure(num=1, figsize=(8, 6))
plt.xlabel('age', size=14)
plt.ylabel('count', size=14)
plt.bar(ages, counts, width=0.5)
plt.show()
```

Скриншоты

В место # было решено использовать обычный счетчик.