УДК 004

Разработка парсера постов во ВКонтакте

Андриенко Иван Сергеевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье описывается процесс разработки парсера для сбора информации о постах и изображениях из групп ВКонтакте. Для разработки использовался Python и модуль requests для выполнения запросов к API ВКонтакте. Результаты исследования показывают, что разработка парсера постов во ВКонтакте является эффективным способом сбора информации, которую можно использоваться в различных целях.
Ключевые слова: vk api, Python, парсер, ВКонтакте, requests

Development of a post parser in VKontakte

Andrienko Ivan Sergeevich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article describes the process of developing a parser for collecting information about posts and images from VKontakte groups. Python and the vk_api library were used for development. The results of the study show that the development of a post parser in VKontakte is an effective way to collect information that can be used for various purposes.
Keywords: vk api, Python, parser, vkontakte, requests

1 Введение

1.1 Актуальность

В современном информационном обществе социальные сети играют важную роль в обмене информацией, коммуникации и создании контента. ВКонтакте, как одна из самых популярных социальных платформ, предлагает огромное количество данных в виде постов и изображений, которые могут быть ценными для анализа и исследований.

Разработка парсера для извлечения информации из ВКонтакте является актуальной темой в сфере аналитики данных и исследования социальных медиа. Парсинг данных позволяет получить доступ к информации, которая может быть использована для различных целей, например, выявление трендов или создание рекомендательных систем. Автоматизированный сбор постов и изображений может быть полезен для маркетинговых исследований, конкурентного анализа, создания персонализированных рекламных кампаний,

мониторинга обсуждений и мнений пользователей, а также для создания приложений, основанных на контенте из ВКонтакте.

1.2 Обзор исследований

К.А. Вронский, Е.А. Шеленок рассмотрели задачу поиска автоматизированных способов сбора информации с социальной сети «ВКонтакте». В качестве ее решения предлагается использование парсинга на основе извлечения текстовых данных с HTML-страницы и взаимодействие с инструментарием VKAPI [1]. Г.С. Сейдаметов, Ш.М. Усеинов, описали возможность загрузки фотографий через api vk на языке программирования php. Взаимодействие с API VK для загрузки медиа-контента на сервера VK. В статье представлена последовательность реализации работающего кода, который можно будет использовать для добавления фотографий на стену, в альбомы или же в диалоги [2]. В своей работе Д.Р. Вихляев и В.А. Глаголев описывают реализацию парсинга группы или сообщества в социальной сети «ВКонтакте». Программа реализована с помощью методов Api vk, и библиотеки jQuery. Результатом исследования станет готовая программа с считывающая данные о новостях сообщества [3]. Низомутдинов Б.А., Филатова О.Г. представили сравнительный анализ двух официальных источников информации, на примере группы в социальной сети ВКонтакте и Telegram. Использованы автоматизированные методы сбора и обработки комментариев. Использовались API ВКонтакте и библиотека Telescrape для парсинга данных из Telegram [4]. В статье Е.С. Комарова исследуется парсинг данных как новый инструмент маркетингового анализа собственных данных в социальных сетях. Освещается практическое применение парсинга в социальной сети «ВКонтакте» на основе публикаций, размещенных на стене сообщества. Исследование парсинга «ВКонтакте» ведется через рассмотрение таких тенденций, как изменение структуры посещения социальных сетей пользователем [5].

1.3 Цель исследования

Целью данного исследования является разработка эффективного парсера для извлечения постов и изображений из групп ВКонтакте.

2 Материалы и методы

В Процессе создания парсера использовалась среда программирования PyCharm Community Edition 2022.1.3, язык программирования python и модуль requests для выполнения запросов к API ВКонтакте.

3 Результаты и обсуждение

VK (ВКонтакте) – одна из крупнейших социальных сетей в мире, с миллионами активных пользователей и разнообразными функциями. Однако, ВКонтакте не только является платформой для социального взаимодействия, но и предлагает разработчикам широкие возможности для создания собственных приложений и интеграции с социальной сетью. Сеть

предоставляет набор API (интерфейсов программирования приложений), который позволяет разработчикам получать доступ к данным и функциональности VK для создания различных приложений, ботов, парсеров и других проектов. Это открывает двери для широкого спектра возможностей в области разработки и интеграции.

Прежде чем начать сбор данных, необходимо получить авторизационный токен, который будет использоваться для доступа к API ВКонтакте. Токен предоставляет разрешение на сбор данных с аккаунтов и групп.

Для получения токена обратимся к сайту разработчиков ВКонтакте[6] и зарегистрируем приложение (рис.1).

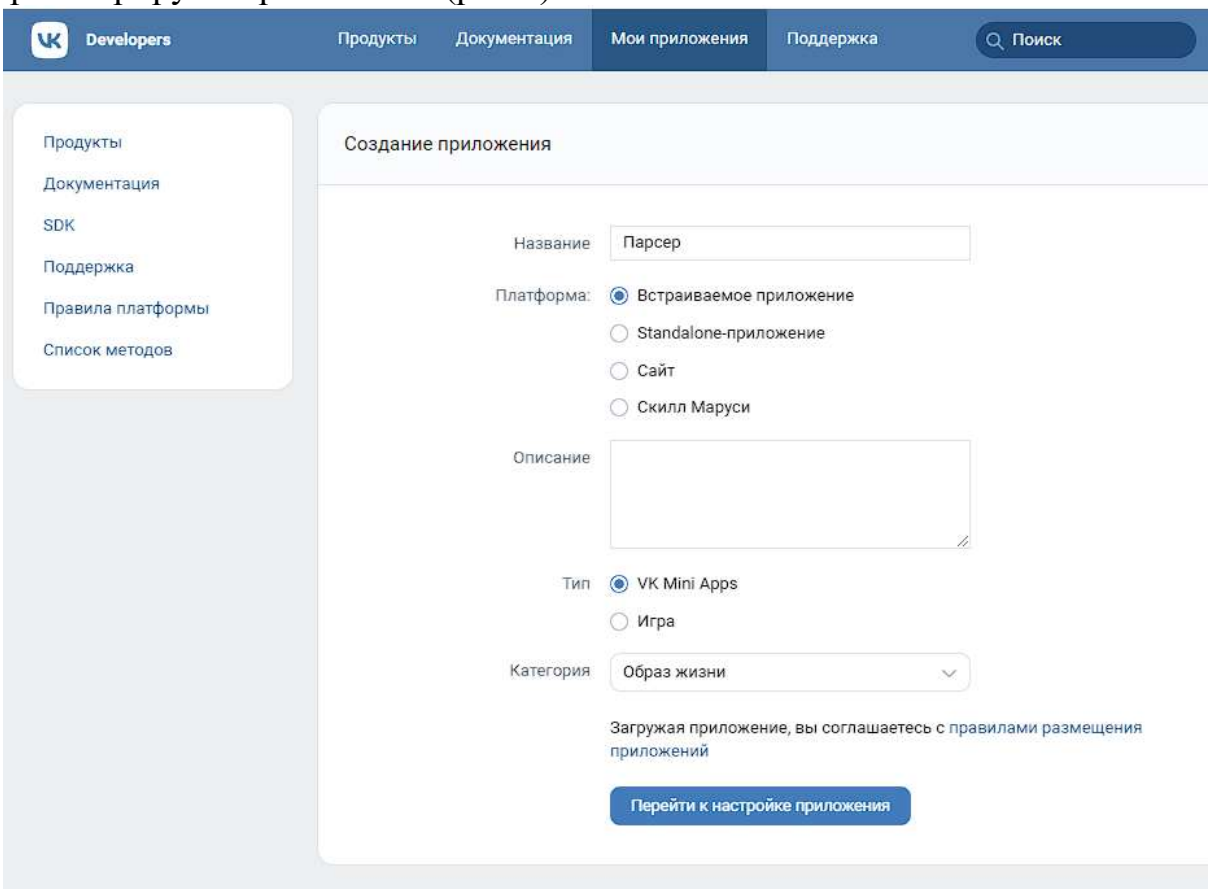

Рисунок 1 – Создание приложения для получения токена

После регистрации выдают уникальный идентификатор – id приложения, защищённый ключ и сервисный ключ доступа (рис. 2). В данном случае необходим только сервисный ключ доступа, он будет являться ключевым элементом парсера, позволяя авторизоваться и получать данные из ВКонтакте. Важно сохранить этот токен в безопасности и не делиться им с другими людьми, чтобы избежать злоупотребления или несанкционированного доступа к данным.

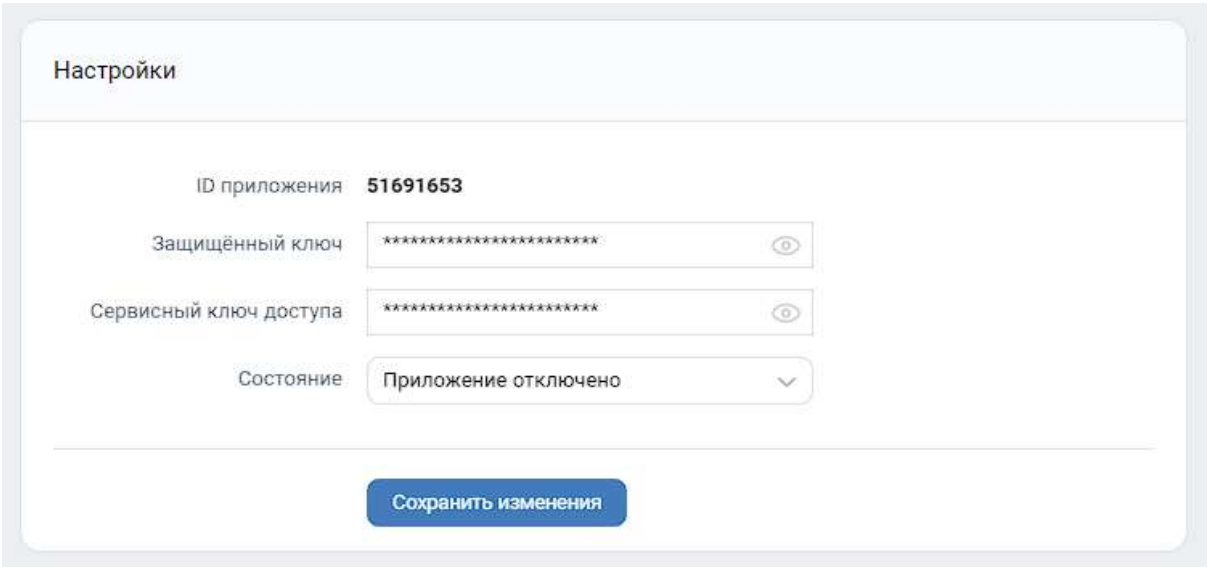

Рисунок 2 – Получение ключа доступа

Теперь, когда есть токен, все готово для разработки парсера и извлечению постов и изображений из ВКонтакте. Импортируем необходимые модули и библиотеки (рис. 3). Функция «urlparse» используется для разбора URL-адресов и извлечения информации о них, такой как протокол, хост, путь и другие компоненты.

```
import requests
import json
import os
from urllib.parse import urlparse
```

Рисунок 3 – Импорт библиотек и модулей

Далее определяем идентификатор группы (group_id), который идентифицирует конкретную группу ВКонтакте, из которой будут извлекаться посты. Затем, для выполнения запросов к API ВКонтакте, указываем ранее полученный токен доступа (access_token), который предоставляет необходимые разрешения для получения данных из группы. Затем задаем количество постов, которые необходимо спарсить, с помощью переменной (post_count). Это позволяет контролировать количество получаемых постов и настраивать парсер под наши нужды. Далее вызываем функцию get_group_posts(), передавая ей идентификатор группы, токен доступа и количество постов. Эта функция обращается к API ВКонтакте и возвращает список постов из указанной группы. Полученный список постов передается в функцию parse_posts() (рис. 4).

```
def main():
    group_id = '-63311869'
    access_token = 'dc3f5a63dc3f5a63dc3f5a63d5df2b9ae6ddc3fdc3f5a63b8a251625[redacted]'
    post_count = 1

    posts = get_group_posts(group_id, access_token, post_count)

    parse_posts(posts)
```

Рисунок 4 – Создание переменных для парсера

Для парсинга постов необходимо получить информацию о постах. Функция get_group_posts() выполняет запрос к API VK для получения списка постов из заданной группы. Внутри функции формируется URL-запрос к методу wall.get API VK, указывая идентификатор группы, количество постов, токен доступа и версию API. Затем выполняется GET-запрос по указанному URL и полученный ответ преобразуется в формат JSON. Далее, производится проверка наличия ошибки в ответе. Если в ответе присутствует поле 'error', извлекается сообщение об ошибке и выбрасывается исключение с соответствующим сообщением. Если ошибки отсутствуют, извлекается список постов из поля 'items' в ответе и возвращается в качестве результата функции (рис. 5).

```
def get_group_posts(group_id, access_token, post_count):
    url = f'https://api.vk.com/method/wall.get?owner_id={group_id}&count={post_count}&access_token={access_token}&v=5.131'
    response = requests.get(url)
    data = json.loads(response.text)

    if 'error' in data:
        error_message = data['error']['error_msg']
        raise Exception(f'Error: {error_message}')

    posts = data['response']['items']
    return posts
```

Рисунок 5 – Запрос для получения постов

Пропишем код, который отвечает за сбор текста и изображений с поста. В функции parse_posts() реализована следующая логика:

1. Для каждого поста в списке posts выполняются следующие действия:
2. Получаем текст поста и сохраняем его в переменную text.
3. Выводим текст поста на экран.
4. Проверяем наличие вложений в посте.
5. Если вложение является фотографией, выполняем следующие шаги:
   - Получаем список доступных размеров фотографии из вложения.
   - Определяем фотографию с наибольшим размером по ширине.
   - Получаем URL фотографии.
   - Извлекаем имя файла из URL фотографии.
   - Выполняем загрузку изображения.

Процесс повторяется для каждого вложения в посте, если они имеются. Таким образом, функция позволяет собрать текст и изображения с каждого поста и выполнить необходимые операции с ними (рис. 6).

```
34
35  def parse_posts(posts):
36      for post in posts:
37          text = post['text']
38          print(f'Пост: {text}')
39
40          attachments = post.get('attachments', [])
41          for attachment in attachments:
42              if attachment['type'] == 'photo':
43                  photo_sizes = attachment['photo']['sizes']
44                  max_size_photo = max(photo_sizes, key=lambda x: x['width'])
45                  photo_url = max_size_photo['url']
46
47                  parsed_url = urlparse(photo_url)
48                  filename = os.path.basename(parsed_url.path)
49
50                  download_image(photo_url, filename)
51
```

Рисунок 6 – Получение текста и изображений

Осталось скачать изображения. Создадим функцию download_image. Она будет отвечать отвечает за скачивание изображения по заданному URL и его сохранение на диск. Вначале код проверяет, существует ли папка "images" для сохранения изображений. Если папка не существует, она создается с помощью os.makedirs(). Затем изображение загружается с помощью запроса requests.get(url), а полученные данные записываются в файл с помощью file.write(response.content). После успешного сохранения файла выводится сообщение о пути, по которому изображение было сохранено (рис. 7).

```python
def download_image(url, filename):
    if not os.path.exists('images'):
        os.makedirs('images')

    image_path = os.path.join('images', filename)
    with open(image_path, 'wb') as file:
        response = requests.get(url)
        file.write(response.content)

    print(f'Изображение скачано: {image_path}')


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

Рисунок 7 – Скачивание и сохранение изображений

Запускаем программу (рис. 8, 9)

```
C:\Users\andrv\PycharmProjects\parserVK\venv\Scripts\python.exe C:/Users/andrv/PycharmProjects/parserVK/main.py
Пост: Поступай на бюджет в магистратуру ПГУ имени Шолом-Алейхема!

В этом году у каждого, имеющего диплом бакалавра или специалиста, есть возможность поступить еще и в магистратуру на любой профиль

Эколог может за 2 года получить диплом магистра по Экономике, психолог - диплом электроэнергетика и т.д.

Причем, у нас в магистратуре и в очной, и в заочной формах имеются бюджетные места!

Читай подробнее — https://pgusa.ru/news/postupay-na-byudzhet-v-magistraturu-pgu-imeni-sholom-aleynema—

#ПГУимША #ЕдинственныйРегиональный
Изображение скачано: images\e_VFfo1cSZ8.jpg
```

Рисунок 8 – Вывод текста поста

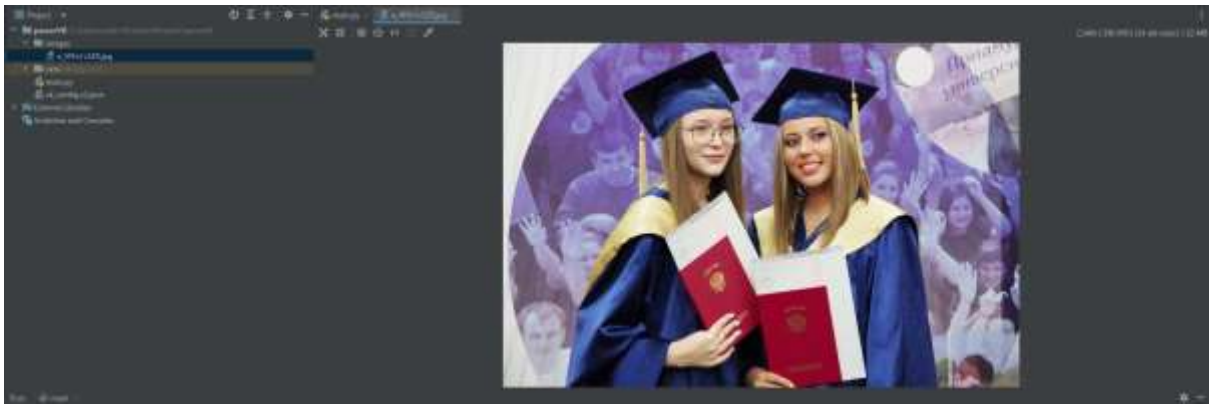

Рисунок 9 – Получение изображения в папку

Выводы

В данной работе был разработан парсер для извлечения постов и изображений из группы ВКонтакте. Используя API ВКонтакте и модуль

requests, реализована возможность получения доступа к постам, извлечения текстов постов и загрузки изображений. Парсер представляет ценность для анализа данных, исследований и автоматизации задач, связанных с сбором информации во ВКонтакте.

Библиографический список

1. Вронский К.А., Шеленок Е.А. Методы автоматизированного сбора информации с социальной сети «ВКонтакте» // Информационные технологии XXI века. Сборник научных трудов. Редакционная коллегия: ответственный редактор В.В. Воронин. Хабаровск, 2021. С. 389-393.
2. Сейдаметов Г.С., Усеинов Ш.М. Загрузка фотографий через api vk на языке программирования php // Информационно-компьютерные технологии в экономике, образовании и социальной сфере. 2018. № 2 (20). С. 35-41.
3. Вихляев Д.Р., Глаголев В.А. Парсинг данных сообщества "Вконтакте" с помощью VK API // Постулат. 2021. № 10 (72).
4. Низомутдинов Б.А., Филатова О.Г. Тестирование методов обработки комментариев из telegram-каналов и пабликов вконтакте для анализа социальных медиа // International Journal of Open Information Technologies. 2023. Т. 11. № 5. С. 137-145.
5. Комарова Е.С. Новый инструментарий маркетингового анализа собственных данных в социальных сетях // Современные научные исследования актуальные вопросы, достижения и инновации. Сборник статей XXX Международной научно-практической конференции. Пенза, 2022. С. 103-108.
6. VK для разработчиков URL: https://dev.vk.com (дата обращения: 28.06.2023)