

Работа с сетью, практика

Юрий Литвинов

y.litvinov@spbu.ru

В этот раз будет практическое занятие на использование веб-сервисов и работу с протоколом авторизации OAuth — написание консольного клиента для ВКонтакте с использованием VK REST API. Работа предполагается в командах по 3-4 человека, где большая часть участников занимается чтением документации. Хотя бы у одного члена команды должен быть аккаунт в VK.

Собственно, вся необходимая для практики информация есть в официальной документации, <https://vk.com/dev/manuals>. Задача будет состоять из трёх частей, от простого к сложному: сначала получение открытой информации, затем аутентификация и получение списка контактов онлайн, затем (если успеем, что вряд ли) загрузка изображения на сервера VK и выставление его как аватара сообщества.

1. Получение открытой информации о пользователе

Первая задача — научиться получать открытую информацию о пользователе по его Id. Для этого надо, во-первых, зарегистрировать своё приложение в VK API. Это делается через страницу VK, меню слева, «Управление», «Мои приложения», «Создать». Выбираете при создании Standalone-приложение. Дальше в его настройках можно увидеть сервисный ключ доступа, защищённый ключ и т.д., это потом потребуется для аутентификации.

Дальше уже в среде разработки создаём консольное приложение, вспоминаем, как делать HTTP-запросы из .NET-приложений, и параллельно внимательно читаем https://vk.com/dev/first_guide. Там, чтобы было не скучно, написана неправда, запрос `users.get` не исполнится просто так, он требует сервисного ключа от включённого в настройках VK приложения. Сервисный ключ надо передавать как параметр `access_token`. Наконец, полезно почитать про сам нужный нам метод API: <https://vk.com/dev/users.get> и про работу с разного сорта ключами (с сервисным ключом всё просто, но дальше нам потребуется честная аутентификация): https://vk.com/dev/access_token.

Отвечать сервер будет JSON-документами, вот справка по System.Text.Json касательно десериализации (высокоуровневой) документов: <https://learn.microsoft.com/en-us/dotnet/standard/serialization/system-text-json/deserialization>. Чтобы оно работало, надо в коде на C# описать классы-данные, которые будут соответствовать полям JSON-объектов (то есть иметь свойства с правильными именами и правильными типами). См. также <https://learn.microsoft.com/en-us/dotnet/standard/serialization/system-text-json/character-casing>.

Если всё пойдёт хорошо, в результате у вас должно получиться консольное приложение, куда можно ввести id пользователя и получить его имя-фамилию и другие поля, кото-

рые сочтёте интересными. Сервисный ключ стоит вводить как параметр командной строки или читать из файла, только не выкладывайте его на GitHub (в том числе и в конфиге запуска).

2. Авторизация, список контактов онлайн

Часть вторая этого задания требует получения списка контактов текущего пользователя, которые в данный момент находятся онлайн. Это уже нельзя сделать по сервисному ключу, требуется аутентифицировать конкретного пользователя (чтобы VK знал, чьи контакты ему выдавать).

Для этого надо почитать документацию по Implicit Flow: https://vk.com/dev/implicit_flow_user. Вам потребуется открыть страницу браузера на правильный URL, ввести свои логин и пароль, после чего браузер редиректнут на «страницу», содержащую access_token. Его-то и надо использовать как access_token для каждого следующего запроса. Его можно просто просить пользователя скопировать в окно консоли из адресной строки браузера, ничего хитрее придумывать не нужно (и на паре всё равно не успеете, скорее всего, так что проходить аутентификацию надо будет каждый раз заново)¹. Нужный метод API, который, собственно, вернёт список контактов, надо найти самостоятельно.

Вот как запустить браузер из .NET под Windows:

```
authString = authString.Replace("&", "^&");
Process.Start(new ProcessStartInfo(
    "cmd",
    $"/c start {authString}")
{ CreateNoWindow = true });
```

3. Поменяем аватарку

И последняя часть задания — поменять аватарку текущему пользователю. Тут придётся хотя бы на базовом уровне разобраться с тем, что такое MIME, и как делать POST-запросы. Тут потребуется сделать три запроса:

- получить адрес, по которому надо загрузить фото: <https://vk.com/dev/photos.getOwnerPhotoUploadServer>;
- собственно загрузить фото: https://vk.com/dev/upload_files;
- выставить загруженное фото как аватар: <https://vk.com/dev/photos.saveOwnerPhoto>.

Тут с деталями придётся разобраться самостоятельно, но раз аутентификация уже реализована, самое сложное позади. Вот как сформировать multipart/form-data в формате, который поймёт сервер вконтакта:

¹ По-хорошему, надо открыть HttpListener на какой-нибудь порт на localhost и указать redirect_uri=http://localhost:<ваш порт>/ в параметре redirect_uri запроса авторизации, и считать из URL входящего запроса код авторизации (обратите внимание, сервер отвечает браузеру «redirect» с указанием того URL, который вы ему передали, браузер честно делает запрос — не сам сервер идёт на localhost, конечно).

```
var form = new MultipartFormDataContent
{
    { new ByteArrayContent(File.ReadAllBytes("cthulhu.png"))
      , "photo", "cthulhu.png" }
};
```