

**Московский авиационный институт**

**(национальный исследовательский университет)**

Институт №8 «Информационные технологии и прикладная математика»

Кафедра 806 «Вычислительная математика и программирование»

Дисциплина «Операционные системы»

## **Курсовой проект**

Тема: Бот ВК

Студент: Глушатов И.С. Группа:

М8О-207Б-19 Преподаватель:

Миронов Е. С. Дата:

Оценка:

Москва, 2020

## Содержание

1. Задача и цель работы.....	3
2. Реализация.....	4
3. Листинг программы.....	5
4. Тесты и протокол исполнения.....	11
5. Вывод.....	14
6. Список литературы.....	14

## Задача и цель работы

**Цель работы:** закрепить навыки разработки динамических библиотек на языке C++ на операционной системе Windows. Научиться работать с python библиотекой vk\_api. Научится связывать динамические библиотеки на языке C++ с кодом на python.

**Задача:** создать бота ВК, привязанного к определенной группе, который будет принимать команды и выдавать на них ответ. Весь функционал для работы с файловой системой должен быть реализован в динамической библиотеке.

Команды:

1. get file `path` - выдает содержимое файла
2. get files `path` - выдает список всех файлов и каталогов в заданном каталоге
3. delete `path` - удаляет файл
4. get size `path` - выдает размер файла в байтах

Поиск по файловой системе будет производиться на том компьютере, на котором запущен бот.

## Реализация

Основная программа на языке python авторизирует бота и в бесконечном цикле начинает принимать сообщения. Каждое полученное сообщение содержит id отправителя и текст. Эта информация отправляется объекту ChatBot на обработку и от него получаем ответ, который далее отправляем тому пользователю, от которого получили запрос.

Обработка запроса производится в методе new\_message с помощью еще одной библиотеки ctypes, позволяющей подключать dll библиотеки и работать с её функциями. На этапе разработки возникли большие трудности с типами переменных, так как в python они не явные и передача аргументов с последующим возвратом значений требовали аккуратности в этом деле. Большие сложности возникли с кодировками символов, а именно русским алфавитом. В итоге получилось добиться того, чтобы можно было обращаться к папкам и файлам с русскими именами.

Dll библиотека написана на языке C++ и выполняет все функции, связанные с командами бота. Для работы с каталогами пришлось компилировать под стандарт 2017 года, так как только начиная с него была введена библиотека <filesystem>.

Не могут выдаваться содержимое пустых файлов и каталогов, и, если размер файла превышает 4Кб или в каталоге очень много файлов и других каталогов – бросается исключение и выдается ошибка. Эти ограничения связаны с тем, что в ВК нельзя отправлять пустые сообщения, либо сообщения с большим количеством символов (во втором случае оно может биться на несколько сообщений, но я этого не реализовал).

## Листинг программы

### VKBot\_script.py

```
import vk_api
from vk_api.longpoll import VkLongPoll, VkEventType
import datetime
import time
from VKBot import ChatBot
from logpas import login, password, token

if __name__ == "__main__":

    longpoll = vk_api.VkApi(login, password) # авторизация аккаунта
    longpoll.auth()
    vk_session = vk_api.VkApi(token=token) # привязка к группе

    longpoll = VkLongPoll(vk_session) # создание сервера

    print("Starting...")
    while True:
        for event in longpoll.listen():
            if event.type == VkEventType.MESSAGE_NEW:
                if event.from_user and not (event.from_me):

                    print ("Время сообщения: " +
                           str(datetime.datetime.now().strftime("%d-%m-%Y
                           %H:%M")))
                    print ("Текст сообщения: " + event.text)
                    print ("ID пользователя: " + str(event.user_id))

                    bot = ChatBot(event.user_id, vk_session)

                    try:
                        vk_session.method('messages.send',
                                           bot.new_message(event.text))
                    except:
                        vk_session.method('messages.send',
                                           bot.new_message("ERROR: big file or something else", 1))
```

## VKBot.py

```
import bs4
from bs4 import BeautifulSoup
import requests
import random
from random import randint
import vk_api
from vk_api.longpoll import VkLongPoll, VkEventType
from vk_api.upload import VkUpload
import json
from io import BytesIO
import ctypes

class ChatBot():

    def __init__(self, user_id, vk):
        self._vk = vk
        self._user_id = user_id
        self._user_name = self._get_user_name_from_vk_id(user_id)
        self._commands = [['привет', 'хай', 'хэлоу', 'здорово', 'здравствуйте', 'здравствуй'],
                           ['get file', 'get files'], ['delete'], ['get size']]

        self.file_library = ctypes.CDLL("C:\\Users\\Igor\\Desktop\\KP OS\\library.dll")

    def _get_user_name_from_vk_id(self, user_id):
        request = requests.get("https://vk.com/id"+str(user_id))
        bs = bs4.BeautifulSoup(request.text, "html.parser")
        user_name = self._get_last_first_name(bs.findAll("title")[0])
        return user_name.split()[0]

    def new_message(self, message, echo = 0):

        word_delete_list = message.split(' ', maxsplit = 1)
        word_list = message.split(' ', maxsplit = 2)

        if echo == 1:
            return {'user_id': self._user_id,
                    'message': message,
                    'random_id': randint(-100000, 100000)}

        #Приветствие
        if message.lower() in self._commands[0]:
            a = ("Привет, " + self._user_name + "!")
            return {'user_id': self._user_id,
```

```

        'message': a,
        'random_id': randint(-100000, 100000)}

elif len(word_list) < 2:
    return {'user_id': self._user_id,
            'message': "Не понимаю о чем ты...",
            'random_id': randint(-100000, 100000)}

#delete path
elif self._commands[3][0] == word_delete_list[0]:
    df = self.file_library.delete_file
    df.restype = ctypes.c_bool
    df.argtypes = [ctypes.c_wchar_p]

    if df(ctypes.c_wchar_p(word_delete_list[1])) == True:
        return {'user_id': self._user_id,
                'message': "Файл успешно удалился",
                'random_id': randint(-100000, 100000)}
    else :
        return {'user_id': self._user_id,
                'message': "Файл не смог удалиться",
                'random_id': randint(-100000, 100000)}

#get file path
elif self._commands[1][0] == word_list[0] + ' ' + word_list[1]:
    gfas = self.file_library.get_file
    gfas.restype = ctypes.c_char_p
    gfas.argtypes = [ctypes.c_wchar_p]
    a = gfas(ctypes.c_wchar_p(word_list[2]))
    return {'user_id': self._user_id,
            'message': a.decode('utf8'),
            'random_id': randint(-100000, 100000)}

#get files path
elif self._commands[2][0] == word_list[0] + ' ' + word_list[1]:
    gfsas = self.file_library.get_files
    gfsas.restype = ctypes.c_char_p
    gfsas.argtypes = [ctypes.c_wchar_p]
    a = gfsas(ctypes.c_wchar_p(word_list[2]))
    return {'user_id': self._user_id,
            'message': a.decode('utf8'),

```

```

        'random_id': randint(-100000, 100000)})

    #get size path
    elif self._commands[4][0] == word_list[0] + ' ' + word_list[1]:
        gsai = self.file_library.get_file_size
        gsai.restype = ctypes.c_longlong
        gsai.argtypes = [ctypes.c_wchar_p]
        a = gsai(ctypes.c_wchar_p(word_list[2]))
        return {'user_id': self._user_id,
                'message': str(a),
                'random_id': randint(-100000, 100000)}

    else:
        return {'user_id': self._user_id,
                'message': "Не понимаю о чем ты...",
                'random_id': randint(-100000, 100000)}

    @staticmethod
    def _get_last_first_name(legs_str):
        res = ""
        for i in legs_str.get_text():
            if i == ' ':
                break
            else:
                res+=i
        return res

```

## library.cpp

```

#include <string>
#include <stdio.h>
#include <fstream>
#include <iostream>
#include <conio.h>
#include <windows.h>
#include <filesystem>

#define LIBDLL extern "C" __declspec(dllexport)

LIBDLL const char* get_file (const wchar_t* path) {
    try {
        std::ifstream file;
        file.open(path, std::fstream::in);
        if (!file) {
            return "Can't open or not available\n";
        } else {
            std::string result = "";
            char a;

```



```

        while (file.get(a)) {
            result += a;
        }
        file.close();
        char* res = new char[result.length()+1];
        strcpy(res, result.c_str());
        return (const char*) res;
    }
} catch (const std::exception &e) {
    return e.what();
} catch (...) {
    return "Everything is bad :(";
}
return "No such file(";
}

```

```

LIBDLL const char* get_files (const wchar_t* path) {
    SetConsoleOutputCP(CP_UTF8);
    try {
        namespace fs = std::filesystem;
        std::string result = "";
        for (auto & p : fs::directory_iterator(path)) {
            result += p.path().string() + "\n";
        }
        char* res = new char[result.length()+1];
        strcpy(res, result.c_str());
        return (const char*) res;
    } catch (const std::exception &e) {
        return e.what();
    } catch (...) {
        return "Everything is bad :(";
    }
    return "No such directory(";
}

```

```

LIBDLL bool delete_file (const wchar_t* path) {
    try {
        namespace fs = std::filesystem;
        bool ret = fs::remove(fs::path(path));
        return ret;
    } catch (const std::exception &e) {
        return false;
    } catch (...) {
        return false;
    }
}

```

```

    }
    return false;
}

LIBDLL long long get_file_size (const wchar_t* path) {
    try {
        std::ifstream file;
        file.open(path, std::ifstream::ate);
        if (!file) {
            return -1;
        } else {
            return file.tellg();
        }
    } catch (const std::exception &e) {
        return -1;
    } catch (...) {
        return -1;
    }
    return -1;
}

```

## Тесты и протокол исполнения



Игорь 19:50

get size C:\\Users\\Igor\\Desktop\\pl1.pl



Идеи какого-то Олега 19:50

13859



Игорь 19:50

get size C:\\Users\\Igor\\Desktop\\pl.pl



Идеи какого-то Олега 19:50

62



Игорь 19:50

get file C:\\Users\\Igor\\Desktop\\pl.pl



Идеи какого-то Олега 19:50

append([], X, X).

append([A|X], Y, [A|Z]) :- append(X, Y, Z).







**Игорь** 19:49

get files C:\



**Идеи какого-то Олега** 19:49

C:\\$Recycle.Bin  
C:\\$WinREAgent  
C:\bootmgr  
C:\BOOTNXT  
C:\Distr  
C:\Documents and Settings  
C:\DumpStack.log.tmp  
C:\FPC  
C:\Games  
C:\hiberfil.sys  
C:\library.dll  
C:\NVIDIA  
C:\OneDriveTemp  
C:\pagefile.sys  
C:\PerfLogs  
C:\Program Files  
C:\Program Files (x86)  
C:\ProgramData  
C:\Python 3.8  
C:\Recovery  
C:\swapfile.sys  
C:\System Volume Information  
C:\Users  
C:\Windows



**Игорь** 19:49

get files C:\Users\Igor\



**Идеи какого-то Олега** 19:49

C:\Users\Igor\.bash\_history  
C:\Users\Igor\designer  
C:\Users\Igor\gitconfig  
C:\Users\Igor\idlerc  
C:\Users\Igor\3D Objects  
C:\Users\Igor\ansel  
C:\Users\Igor\AppData  
C:\Users\Igor\Application Data  
C:\Users\Igor\Contacts  
C:\Users\Igor\Cookies  
C:\Users\Igor\cpython  
C:\Users\Igor\Desktop  
C:\Users\Igor\Doctor Web  
C:\Users\Igor\Documents  
C:\Users\Igor\Downloads  
C:\Users\Igor\Favorites  
C:\Users\Igor\Intel  
C:\Users\Igor\Links  
C:\Users\Igor\Local Settings  
C:\Users\Igor\Music  
C:\Users\Igor\NetHood  
C:\Users\Igor\NTUSER.DAT  
C:\Users\Igor\ntuser.dat.LOG1  
C:\Users\Igor\ntuser.dat.LOG2  
C:\Users\Igor\NTUSER.DAT{d2f71646-d4ec-11ea-b460-8c53be63bf10}.TM.blf  
C:\Users\Igor\NTUSER.DAT{d2f71646-d4ec-11ea-b460-8c53be63bf10}.TMContainer00000000000000000001.regtrans-ms  
C:\Users\Igor\NTUSER.DAT{d2f71646-d4ec-11ea-b460-8c53be63bf10}.TMContainer00000000000000000002.regtrans-ms  
C:\Users\Igor\ntuser.ini  
C:\Users\Igor\OneDrive  
C:\Users\Igor\Pictures



Напишите сообщение...



## **Выводы**

В данной курсовой работы я познакомился с разработкой чат-ботов для социальной сети ВКонтакте. Создал бота, которого могут использовать любые пользователи сети, когда программа работает. Столкнулся с рядом проблем при реализации. Во-первых, python и C++ плохо взаимодействуют в плане кодировок. Однако после использования типа `whar_t` все получилось. Второй проблемой оказалась сложность подключения динамической библиотеки к python коду. Для этого я использовал дополнительную библиотеку `ctypes`, однако я не уверен, что это единственно правильный вариант. А примеров ее использования со строковыми переменными очень мало, поэтому пришлось много экспериментировать. К тому же в программе еще многое что можно дорабатывать. Выдавать осмысленные ответы на пустые файлы и каталоги, а также “резать” сообщения, если их размер очень большой.

## **Список литературы**

1. Поисковик Google [электронный ресурс] URL: <https://google.com/> (дата обращения: 26.12.2020)
2. Документация по `vk_api` [электронный ресурс] URL: <https://vk-api.readthedocs.io/en/latest/> (дата обращения: 26.12.2020)
3. Документация по `ctypes` [электронный ресурс] URL: <https://docs.python.org/3/library/ctypes.html> (дата обращения: 25.12.2020)
4. Связка `dll` & `python` [электронный ресурс] URL: <https://habr.com/ru/post/499152/> (дата обращения: 25.12.2020)