БГУИР

# Кафедра ЭВМ

### Отчет по лабораторной работе № 4

**Тема: «Веб-камера»**

Выполнил:

студент группы 130501 Гнездилов А.М.

Проверила:

ассистент Игнатович А.О.

#### Минск 2023

1. **ЦЕЛЬ РАБОТЫ**

Вывести информацию об установленной веб-камере.

Осуществить захват изображения (фото и видео) с последующим сохранением в файл.

По вариантам:

- предусмотреть скрытый вариант фотонаблюдения, когда на мониторе и на панели задач не отображается информация о Вашем работающем приложении;

- предусмотреть скрытый вариант видеонаблюдения, когда на мониторе и на панели задач не отображается информация о Вашем работающем приложении.

1. **ЛИСТИНГ КОДА**

**Libraries.h**

#include <iostream>

#include "conio.h"

#include <string>

#include <Windows.h>

#include <Vfw.h>

#include <dshow.h>

#pragma comment(lib, "vfw32")

#pragma comment(lib, "strmiids")

#pragma comment(lib, "setupapi.lib")

#include <setupapi.h>

using namespace std;

**WEB.h**

#include "libraries.h"

class WEB

{

private:

HDEVINFO DeviceDisckriptor;

SP\_DEVINFO\_DATA DeviceInformation;

public:

WEB();

~WEB();

void GetData();

void TakeVideo();

};

**WEB.cpp**

#include "WEB.h"

WEB::WEB()

{

DeviceDisckriptor = SetupDiGetClassDevsW(NULL, TEXT("USB"), NULL, DIGCF\_PRESENT | DIGCF\_ALLCLASSES);

DeviceInformation.cbSize = sizeof(SP\_DEVINFO\_DATA);

SetupDiEnumDeviceInfo(DeviceDisckriptor, 4, &DeviceInformation);

}

WEB::~WEB()

{

SetupDiDestroyDeviceInfoList(DeviceDisckriptor);

}

void WEB::GetData()

{

TCHAR buffer[1024];

SetupDiGetDeviceRegistryPropertyW(DeviceDisckriptor, &DeviceInformation, SPDRP\_FRIENDLYNAME, NULL, (BYTE\*)buffer, 1024, NULL);

cout << "Имя: "; wcout << buffer << endl;

SetupDiGetDeviceRegistryPropertyW(DeviceDisckriptor, &DeviceInformation, SPDRP\_HARDWAREID, NULL, (BYTE\*)buffer, 1024, NULL);

cout << "VID: "; wcout << wstring{ buffer }.substr(wstring{ buffer }.find(L"VID\_") + 4, 4) << endl;

cout << "PID: "; wcout << wstring{ buffer }.substr(wstring{ buffer }.find(L"PID\_") + 4, 4) << endl;

}

void WEB::TakeVideo()

{

CoInitializeEx(NULL, COINIT\_APARTMENTTHREADED); //Инициализация среды COM

ICaptureGraphBuilder2\* builder = nullptr; //указатели на создание графом захвата видео

IGraphBuilder\* graph = nullptr; //указатели на управление графом захвата видео

/\*Создание экземпляра объекта ICaptureGraphBuilder2(предоставляет методы для построения и управления графов видоезахвата\*/

CoCreateInstance(CLSID\_CaptureGraphBuilder2, NULL, CLSCTX\_INPROC\_SERVER, IID\_ICaptureGraphBuilder2, reinterpret\_cast<void\*\*>(&builder));

CoCreateInstance(CLSID\_FilterGraph, 0, CLSCTX\_INPROC\_SERVER, IID\_IGraphBuilder, reinterpret\_cast<void\*\*>(&graph));

/\*Установка связи между builder и graph\*/

builder->SetFiltergraph(graph);

ICreateDevEnum\* dev\_enum = nullptr; //перечисление устройств

CoCreateInstance(CLSID\_SystemDeviceEnum, NULL, CLSCTX\_INPROC\_SERVER, IID\_PPV\_ARGS(&dev\_enum));

IEnumMoniker\* enum\_moniker;

/\*перечисление устройств\*/

dev\_enum->CreateClassEnumerator(CLSID\_VideoInputDeviceCategory, &enum\_moniker, 0);

dev\_enum->Release();

IMoniker\* moniker = nullptr;

IBaseFilter\* filter = nullptr;

/\*получение информации найденого устройства\*/

enum\_moniker->Next(1, &moniker, NULL);

moniker->BindToObject(0, 0, IID\_IBaseFilter, reinterpret\_cast<void\*\*>(&filter)); //сохр. в filter

graph->AddFilter(filter, L"Capture Filter"); //filter закидываем в graph с названием

string name\_file = "video.avi";

//создание и перезапись

CreateFileA(name\_file.c\_str(), GENERIC\_READ | GENERIC\_WRITE, FILE\_SHARE\_READ | FILE\_SHARE\_WRITE, NULL, CREATE\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);

//мультиплексирование видео

IBaseFilter\* mux = nullptr;

wstring w\_name\_file(name\_file.begin(), name\_file.end()); //формат в широких символах

builder->SetOutputFileName(&MEDIASUBTYPE\_Avi, w\_name\_file.c\_str(), &mux, NULL); //настройка вывода в формате avi

builder->RenderStream(&PIN\_CATEGORY\_CAPTURE, &MEDIATYPE\_Video, filter, NULL, mux); //настройка потока захвата и записи

mux->Release(); //освобождение ресура

IMediaControl\* control = nullptr;

/\*управление видеовоспроизведением\*/

graph->QueryInterface(IID\_IMediaControl, reinterpret\_cast<void\*\*>(&control));

system("cls");

int lenght = 0;

cout << "Продолжительность видео: ";

cin >> lenght;

system("cls");

control->Run();

cout << "Идет запись видео... ";

Sleep(lenght \* 1000);

control->Stop();

control->Release();

}

**Main.cpp**

#include "WEB.h"

void menu()

{

cout << endl << "1 - Скрыть окно"

<< endl << "2 - Запись видео"

<< endl << "3 - Показать окно"

<< endl << "0 - Выход"

<< endl;

}

int main() {

ShowWindow(GetConsoleWindow(), SW\_NORMAL);

system("chcp 1251");

WEB CAM;

while (1)

{

system("cls");

CAM.GetData();

menu();

switch (\_getch())

{

case '1':

{

ShowWindow(GetConsoleWindow(), SW\_HIDE);

break;

}

case '2':

{

CAM.TakeVideo();

break;

}

case '3':

{

ShowWindow(GetConsoleWindow(), SW\_NORMAL);

break;

}

case '0':

{

system("cls");

ShowWindow(GetConsoleWindow(), SW\_NORMAL);

cout << "Работа завершена!" << endl;

return 0;

}

}

}

}

1. **ВЫВОД**

Вывели информацию об установленной веб-камере.

Осуществили захват изображения (фото и видео) с последующим сохранением в файл.