Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей Кафедра информатики

Дисциплина: Операционные среды и системное программирование

ОТЧЕТ

к лабораторной работе № 1

на тему

**ОСНОВЫ ПРОГРАММИРОВАНИЯ В WIN 32 API. ОКОННОЕ ПРИЛОЖЕНИЕ WIN 32 С МИНИМАЛЬНОЙ ДОСТАТОЧНОЙ ФУНКЦИОНАЛЬНОСТЬЮ. ОБРАБОТКА ОСНОВНЫХ ОКОННЫХ СООБЩЕНИЙ**

Выполнил:

студент гр. 153503

Вергасов В.М.

Проверил:

Гриценко Н.Ю.

Минск 2023

**СОДЕРЖАНИЕ**

[1 Цели работы 3](#_Toc146883255)

[2 Краткие теоретические сведения 4](#_Toc146883256)

[3 Полученные результаты 8](#_Toc146883257)

[Выводы 9](#_Toc146883258)

[Список литературы 10](#_Toc146883259)

# 1 ЦЕЛИ РАБОТЫ

1 Изучить основы программирования в Win 32 API.

2 Создать оконное приложение Win 32 с минимальной достаточной функциональностью.

3 Реализовать обработку основных оконных сообщений.

4 Разработать оконное приложение, отображает текст, а также после нажатия кнопки Start в панели инструментов, начинает двигаться по окружности. Так же есть кнопка Stop, которая останавливает движение надписи.

**2 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

Win32 API (Application Programming Interface) — это набор функций, предоставляющий низкоуровневый доступ к операционной системе Windows. Он позволяет разработчикам создавать приложения, которые могут использовать все возможности операционной системы.. Win32 API содержит более 3000 функции для реализации всех видов сервисов операционной системы.

Некоторые термины в пользовательской и программной документации Windows в разных контекстах имеют разные значения. Например, слово служба может относиться к вызываемой в операционной системе стандартной подпрограмме, драйверу устройства или к обслуживающему процессу. Что именно означают те или иные термины, показано в следующем списке:

Функции Windows API. Документированные, вызываемые подпрограммы в WindowsAPI. Например, CreateProcess, CreateFile и GetMessage[1].

Оконное приложение — это приложение, которое работает в рамках окна на экране. Это может быть программа, которая отображает текст, графику или другие элементы на экране, и пользователь может взаимодействовать с ними через мышь или клавиатуру.

Основные оконные сообщения — это сообщения, которые операционная система отправляет окну, чтобы сообщить ему о различных событиях, таких как изменение размера окна, перемещение окна, нажатие клавиш на клавиатуре и т.д. Приложение должно обрабатывать эти сообщения, чтобы реагировать на действия пользователя.

Библиотеки DLL (dynamic-link libraries — динамически подключаемые библиотеки). Набор вызываемых подпрограмм, связанных вместе в виде двоичного файла, который может быть загружен в динамическом режиме приложениями, которые используют эти подпрограммы.

# 3 ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ

Результат работы программы после запуска показан на рисунке 3.1.

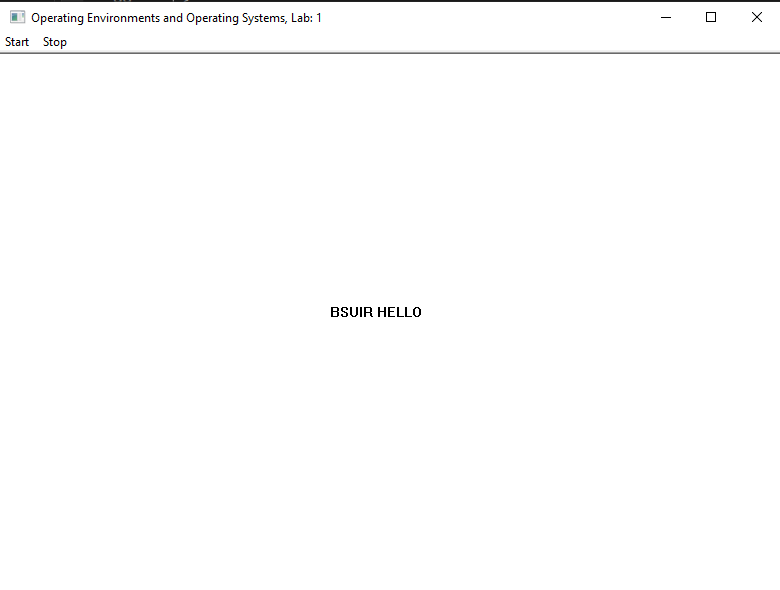


Рисунок 3.1 – Результат работы программы после запуска

После нажатия на кнопку Start надпись начинается двигаться по окружности. Пример одного кадра отображен на рисунке 3.2. Код программы написан в листинге 1.

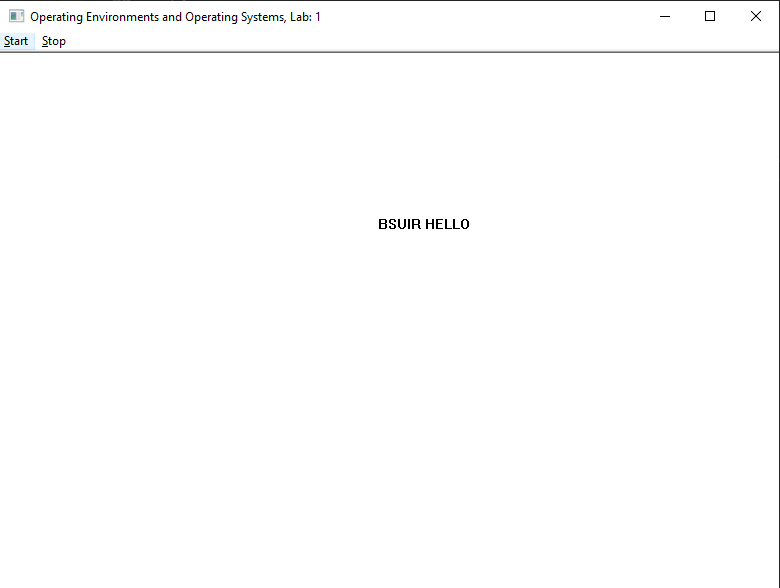


Рисунок 3.2 – Один кадр из всей анимации

Листинг 1 **–** Код исходной программы:

#include <windows.h>

#include <tchar.h>

#include <cmath>

#define IDI\_ICON1 101

#define IDC\_CURSOR1 102

#define IDR\_MENU1 103

#define ID\_START 40001

#define ID\_STOP 40002

#define ID\_TIMER 40003

struct Text {

bool active = false;

int angel = 0;

int shift = 10;

int x = 330, x\_0 = 330;

int y = 250, y\_0 = 250;

LPCWSTR str = L"BSUIR HELLO";

} text;

void Move(RECT \*rect) {

if (text.active) {

text.angel += text.shift;

text.angel %= 360;

text.x = 100 \* cos(text.angel \* (3.14 / 180.0)) + text.x\_0;

text.y = 100 \* sin(text.angel \* (3.14 / 180.0)) + text.y\_0;

}

}

LRESULT CALLBACK WndProc(

HWND hwnd,

UINT msg,

WPARAM wParam,

LPARAM lParam

) {

switch (msg) {

case WM\_CREATE:

{

if (!SetTimer(hwnd, ID\_TIMER, 30, NULL)) {

MessageBox(hwnd, L"Window registration is failed!", L"Error!", MB\_ICONWARNING | MB\_OK);

return false;

}

HDC hdc = GetDC(hwnd);

ReleaseDC(hwnd, hdc);

break;

}

case WM\_COMMAND:

{

switch (LOWORD(wParam)) {

case ID\_START:

{

text.active = true;

break;

}

case ID\_STOP:

{

text.active = false;

break;

}

}

break;

}

case WM\_CLOSE:

{

DestroyWindow(hwnd);

break;

}

case WM\_DESTROY:

{

KillTimer(hwnd, ID\_TIMER);

PostQuitMessage(0);

break;

}

case WM\_TIMER:

{

RECT rect;

HDC hdc = GetDC(hwnd);

GetClientRect(hwnd, &rect);

Move(&rect);

InvalidateRect(hwnd, NULL, TRUE);

ReleaseDC(hwnd, hdc);

break;

}

case WM\_PAINT:

{

PAINTSTRUCT paintStruct;

HDC hdc = BeginPaint(hwnd, &paintStruct);

TextOut(hdc, text.x, text.y, text.str, lstrlen(text.str));

EndPaint(hwnd, &paintStruct);

break;

}

default:

{

return DefWindowProc(hwnd, msg, wParam, lParam);

}

}

}

int WINAPI WinMain(

HINSTANCE hInstance,

HINSTANCE hPrevInstance,

LPSTR lpCmdLine,

int nCmdShow

) {

MSG msg;

HWND hwnd;

WNDCLASSEX wc;

wc.cbSize = sizeof(WNDCLASSEX);

wc.style = 0;

wc.lpfnWndProc = WndProc;

wc.cbClsExtra = 0;

wc.cbWndExtra = 0;

wc.hInstance = hInstance;

wc.hIcon = LoadIcon(GetModuleHandle(NULL), MAKEINTRESOURCE(IDI\_ICON1));

wc.hCursor = LoadCursor(GetModuleHandle(NULL), MAKEINTRESOURCE(IDC\_CURSOR1));

wc.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);

wc.lpszMenuName = MAKEINTRESOURCE(IDR\_MENU1);

wc.lpszClassName = L"VadVergasovWindowClass";

wc.hIconSm = LoadIcon(GetModuleHandle(NULL), MAKEINTRESOURCE(IDI\_ICON1));

if (!RegisterClassEx(&wc)) {

MessageBox(NULL, L"Window registration is failed!", L"Error!", MB\_ICONWARNING | MB\_OK);

return 0;

}

hwnd = CreateWindowEx(

WS\_EX\_CLIENTEDGE,

L"VadVergasovWindowClass",

L"Operating Environments and Operating Systems, Lab: 1",

WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, CW\_USEDEFAULT, 800, 600,

NULL, NULL, hInstance, NULL

);

if (hwnd == NULL) {

MessageBox(NULL, L"Window creation is failed!", L"Error!", MB\_ICONWARNING | MB\_OK);

return 0;

}

HMENU hMenu = CreateMenu();

AppendMenu(hMenu, MF\_STRING, ID\_START, L"&Start");

AppendMenu(hMenu, MF\_STRING, ID\_STOP, L"&Stop");

SetMenu(hwnd, hMenu);

ShowWindow(hwnd, nCmdShow);

UpdateWindow(hwnd);

while (GetMessage(&msg, NULL, 0, 0)) {

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return msg.wParam;

}

# ВЫВОДЫ

В ходе выполнения лабораторной работы были изучены основы программирования в Win 32 API. Как результат, создано оконное приложение с минимальной достаточной функциональностью и реализована обработка основных оконных сообщений.  Разработано оконное приложение, которое позволяет пользователю рисовать и редактировать графические фигуры (круги, прямоугольники) с помощью мыши и клавиш клавиатуры.

# СПИСОК ЛИТЕРАТУРЫ

1. Programming reference for the Win32 API [Электронный ресурс]. – Режим доступа: https://learn.microsoft.com/en-us/windows/win32/api/ – Дата доступа: 16.09.2023.