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

ОТЧЁТ

к лабораторной работе №1

на тему

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

Студент О. Л. Дайнович

Преподаватель Н. Ю. Гриценко

Минск 2023

**СОДЕРЖАНИЕ**

[1 Формулировка задачи 3](#_Toc146874533)

[2 Теоретические сведения 4](#_Toc146874534)

[3 Описание функций программы 5](#_Toc146874535)

[Список использованных источников 7](#_Toc146874536)

[Приложение А (обязательное) Листинг кода 8](#_Toc146874537)

1 ФОРМУЛИРОВКА ЗАДАЧИ

Целью выполнения лабораторной работы является создание оконного приложения на Win32 API, обладающее минимальным функционалом, позволяющим отработать базовые навыки написания программы на Win32 API, таких как обработка оконных сообщений.

В качестве задачи необходимо разработать оконное приложение, которое представляет собой игру “Сапер” с графическим интерфейсом, позволяющую пользователю открывать ячейки поля и помечать мины.

2 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Win32 API (Application Programming Interface) – это набор функций и процедур, предоставляемых операционной системой Windows для разработки приложений на языке программирования C/C++. Оконное приложение Win32 – это приложение, которое состоит из одного или нескольких окон, в которых происходит взаимодействие с пользователем. Для создания окна необходимо зарегистрировать класс окна с помощью функции RegisterClassEx и создать окно с помощью функции CreateWindowEx. Окно может иметь различные свойства, такие как заголовок, размеры, стиль и обработчики сообщений. [1]

Важным аспектом программирования в Win32 API является обработка оконных сообщений. Оконные сообщения – это события, которые происходят в окне, например, нажатие кнопки мыши или клавиши, изменение размера окна и другие действия пользователя. Для обработки оконных сообщений необходимо определить функцию оконной процедуры (WndProc), которая будет вызываться системой при возникновении сообщения. В функции WndProc нужно обрабатывать различные типы сообщений с помощью условных операторов и выполнять соответствующие действия. [2]

3 ОПИСАНИЕ ФУНКЦИЙ ПРОГРАММЫ

Согласно формулировке задачи, были спроектированы следующие функции программы:

–Открытие ячеек поля;

– Пометка мин с помощью “флажков”

*Открытие ячеек поля*

Для открытия ячеек поля игры необходимо нажать на ячейку левой клавишей мыши.

*Пометка мин с помощью “флажков”:*

Для того, чтобы пометить ячейку поля “флажком” (см. рисунок 1), необходимо нажать на данную ячейку правой кнопкой мыши.



Рисунок 1 – Ячейка поля, помеченная “флажком”

Итогом игры может быть победа (см. рисунок 2) либо поражение. Для победы пользователю требуется открыть все ячейки, в которых не находятся мины, а также не нажимать левой кнопкой мыши на ячейки с минами. В случае нажатия пользователем на ячейку с миной он проиграет. Помечать ячейки с минами “флажками” не обязательно для победы.



Рисунок 2 – Победный исход

Ячейки поля с минами помечены знаком “#” (см. рисунок 3), который появляется при нажатии пользователя на данную ячейку.



Рисунок 3 – Поражение

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Build desktop Windows apps using the Win32 API [Электронный ресурс]. – Электронные данные. – Режим доступа: https://learn.microsoft.com/en-us/windows/win32/.

[2] Основные сообщения ОС Windows (Win32 API). Программирование в ОС Windows. Лекция 1. – Электронные данные. – Режим доступа: https://www.youtube.com/watch?v=wTArIolxch0.

ПРИЛОЖЕНИЕ А

(обязательное)

Листинг кода

Листинг 1 – Файл main.cpp

#ifndef UNICODE

#define UNICODE

#endif

#include <windows.h>

#include <windowsx.h>

#include <commctrl.h>

#include <stdlib.h>

#include <time.h>

#include <string>

#include <gdiplus.h>

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

#pragma comment(lib, "comctl32.lib")

using namespace Gdiplus;

#define WINDOW\_WIDTH 500

#define WINDOW\_HEIGHT 500

#define X\_PADDING 140 // отступ слева

#define Y\_PADDING 70 // отступ сверху

#define FIELD\_WIDTH 20

#define FIELD\_HEIGHT 20

#define FIELD\_ROWS 10

#define FIELD\_COLUMNS 10

#define MINES 10 // кол-во мин

#define ID\_BUTTON\_START 1000

#define ID\_BUTTON\_EXIT 993

typedef struct { // клетка поля

bool isMine;

bool isFlag;

bool isOpen;

int minesAround;

} FCell;

FCell field[FIELD\_ROWS][FIELD\_COLUMNS]; // поле

int closedCells; // закрытые ячейки

void new\_game(HWND hwnd);

bool is\_cell\_in\_field(int x, int y);

void open\_fields(int x, int y, HWND hwnd);

LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

LRESULT CALLBACK SubclassWindowProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam, UINT\_PTR uIdSubclass, DWORD\_PTR dwRefData);

int WINAPI wWinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PWSTR pCmdLine, int nCmdShow) // точка входа приложения

{

// Register the window class.

const wchar\_t CLASS\_NAME[] = L"Sample Window Class";

WNDCLASS wc = { };

wc.lpfnWndProc = WindowProc;

wc.hInstance = hInstance;

wc.lpszClassName = CLASS\_NAME;

RegisterClass(&wc);

// Create the window.

HWND hwnd = CreateWindowEx(

0, // Optional window styles. 0 - поведение по умолчанию

CLASS\_NAME, // Window class

L"My Minesweeper", // Window text

WS\_OVERLAPPEDWINDOW, // Window style

// Size and position

CW\_USEDEFAULT, CW\_USEDEFAULT, WINDOW\_WIDTH, WINDOW\_HEIGHT,

NULL, // Parent window

NULL, // Menu

hInstance, // Instance handle

NULL // Additional application data

);

if (hwnd == NULL)

{

return 0;

}

HWND button;

button = CreateWindow(L"BUTTON", L"Start", WS\_VISIBLE | WS\_CHILD, 205, 300, 75, 25, hwnd, (HMENU)ID\_BUTTON\_START, hInstance, NULL);

button = CreateWindow(L"BUTTON", L"Exit", WS\_VISIBLE | WS\_CHILD, 205, 330, 75, 25, hwnd, (HMENU)ID\_BUTTON\_EXIT, hInstance, NULL);

ShowWindow(hwnd, nCmdShow);

// Run the message loop.

MSG msg = { };

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

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return 0;

}

LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)

{

UINT message = uMsg;

switch (uMsg)

{

case WM\_CREATE:

MainWndAddMenus(hwnd);

winRectangle = { 500, 0, 0, 500 };

//textWnd = CreateWindowEx(0, L"STATIC", L"Hello", WS\_VISIBLE | WS\_CHILD | ES\_CENTER, 50, 50, 200, 200, hwnd, NULL, NULL, NULL);

return 0;

case WM\_DESTROY:

PostQuitMessage(0);

return 0;

case WM\_PAINT:

{

HDC hdc = BeginPaint(hwnd, &ps);

EndPaint(hwnd, &ps);

}

return 0;

case WM\_COMMAND:

{

if (wParam == ID\_BUTTON\_START) {

new\_game(hwnd);

return 0;

}

else if (wParam == ID\_BUTTON\_EXIT) {

if (MessageBox(hwnd, L"Really quit?", L"My Minesweeper", MB\_OKCANCEL) == IDOK)

{

DestroyWindow(hwnd);

}

// Else: User canceled. Do nothing.

return 0;

}

else {

HWND button = GetDlgItem(hwnd, LOWORD(wParam));

int buttonId = GetDlgCtrlID(button);

int cellX = (buttonId - 1) % 10;

int cellY = (buttonId - 1) / 10;

if (field[cellX][cellY].isMine) {

SendMessage(button, WM\_SETTEXT, 0, (LPARAM)(L"#"));

if (MessageBox(hwnd, L"You lost!", L"My Minesweeper", MB\_OK) == IDOK)

{

new\_game(hwnd);

}

}

else if (!field[cellX][cellY].isFlag) {

open\_fields(cellX, cellY, hwnd);

}

if (closedCells == 0) {

if (MessageBox(hwnd, L"You won!", L"My Minesweeper", MB\_OK) == IDOK)

{

new\_game(hwnd);

}

}

}

break;

}

return 0;

case WM\_CLOSE:

if (MessageBox(hwnd, L"Really quit?", L"My Minesweeper", MB\_OKCANCEL) == IDOK)

{

DestroyWindow(hwnd);

}

// Else: User canceled. Do nothing.

return 0;

}

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

}

LRESULT CALLBACK SubclassWindowProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam, // обработка правого клика мышки

UINT\_PTR uIdSubclass, DWORD\_PTR dwRefData) {

HWND button = GetDlgItem(hWnd, LOWORD(wParam));

switch (uMsg) {

case WM\_NCDESTROY:

RemoveWindowSubclass(hWnd, SubclassWindowProc, uIdSubclass);

break;

case WM\_RBUTTONDOWN: {

int buttonId = GetDlgCtrlID(hWnd);

int cellX = (buttonId - 1) % 10;

int cellY = (buttonId - 1) / 10;

if (!field[cellX][cellY].isOpen) {

if (!field[cellX][cellY].isFlag) {

SendMessage(hWnd, WM\_SETTEXT, 0, (LPARAM)(L"F"));

field[cellX][cellY].isFlag = true;

}

else {

SendMessage(hWnd, WM\_SETTEXT, 0, (LPARAM)(L""));

field[cellX][cellY].isFlag = false;

}

}

}

return 0;

default:

return DefSubclassProc(hWnd, uMsg, wParam, lParam);

}

return 0;

}

void new\_game(HWND hwnd) { // прорисовка поля игры

srand(time(NULL));

memset(field, 0, sizeof(field));

closedCells = FIELD\_ROWS \* FIELD\_COLUMNS - MINES;

for (int i = 0; i < MINES; i++) { // расстановка мин

int row = rand() % FIELD\_ROWS;

int col = rand() % FIELD\_COLUMNS;

if (field[row][col].isMine)

i--;

else {

field[row][col].isMine = true;

for (int dx = -1; dx < 2; dx++) {

for (int dy = -1; dy < 2; dy++) {

if (is\_cell\_in\_field(row + dx, col + dy))

field[row + dx][col + dy].minesAround++;

}

}

}

}

for (int x = 0; x < FIELD\_ROWS; x++) {

for (int y = 0; y < FIELD\_COLUMNS; y++) {

HWND fieldButton = CreateWindowEx(0, L"BUTTON", L"",

WS\_CHILD | WS\_VISIBLE,

X\_PADDING + x \* FIELD\_WIDTH, Y\_PADDING + y \* FIELD\_HEIGHT, FIELD\_WIDTH, FIELD\_HEIGHT,

hwnd, (HMENU)(x + y \* FIELD\_COLUMNS + 1), NULL, 0);

SetWindowSubclass(fieldButton, SubclassWindowProc, 0, 0);

}

}

}

bool is\_cell\_in\_field(int x, int y) {

return (x >= 0) && (y >= 0) && (x < FIELD\_ROWS) && (y < FIELD\_COLUMNS);

}

void open\_fields(int x, int y, HWND hwnd) { // открытие полей

int buttonId = x + 1 + y \* 10;

HWND button = GetDlgItem(hwnd, buttonId);

field[x][y].isOpen = true;

closedCells--;

if (field[x][y].minesAround) {

SetWindowTextW(button, std::to\_wstring(field[x][y].minesAround).c\_str());

}

else if (field[x][y].minesAround == 0){

SetWindowTextW(button, L".");

for (int dx = -1; dx < 2; dx++) {

for (int dy = -1; dy < 2; dy++) {

if (is\_cell\_in\_field(x + dx, y + dy) && !field[x + dx][y + dy].isOpen && !field[x + dx][y + dy].isFlag)

open\_fields(x + dx, y + dy, hwnd);

}

}

}

}