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 №2

на тему

РАСШИРЕННОЕ ИСПОЛЬЗОВАНИЕ ОКОННОГО ИНТЕРФЕЙСА WIN 32 И GDI+.

Студент О. Л. Дайнович

Преподаватель Н. Ю. Гриценко

Минск 2023

**СОДЕРЖАНИЕ**

[1 Формулировка задачи 3](#_Toc146874412)

[2 Теоретические сведения 4](#_Toc146874413)

[3 Описание функций программы 5](#_Toc146874414)

[Список использованных источников 7](#_Toc146874415)

[Приложение А (обязательное) Листинг кода 8](#_Toc146874416)

1 ФОРМУЛИРОВКА ЗАДАЧИ

Целью выполнения лабораторной работы является освоение расширенное использование оконного интерфейса Win32 и GDI (Graphics Device Interface) для создания сложных изображений, работы с элементами управления и обработки различных сообщений. Изучить механизм перехвата сообщений с использованием WinHook. Изучить и применить GDI для создания сложных изображений, включая рисование графических примитивов, работу с кистями и шрифтами. Настроить обработку различных сообщений от элементов управления и окна, включая сообщения о событиях мыши, клавиатуры и других пользовательских действий. [1]

2 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

WinAPI (Windows API) и GDI+ (Graphics Device Interface Plus) являются двумя ключевыми компонентами для разработки приложений под операционные системы Windows. [3]

WinAPI представляет собой набор функций, структур и констант, предоставляемых операционной системой Windows для создания приложений с графическим интерфейсом пользователя. WinAPI обеспечивает доступ к функциональности операционной системы, такой как создание и управление окнами, обработка сообщений, работа с файлами и директориями, управление ресурсами и многое другое.

В WinAPI используется процедурная модель программирования, где приложение основывается на обработке сообщений, которые поступают от операционной системы и пользовательских действий. Программист создает окна, регистрирует обработчики сообщений и реагирует на события, происходящие в приложении.

WinAPI предоставляет множество функций для работы с окнами, элементами управления, рисованием, вводом-выводом, многопоточностью и другими аспектами разработки приложений под Windows.

GDI+ является надстройкой над GDI (Graphics Device Interface) и предоставляет расширенные возможности для работы с графикой и рисования в Windows. GDI+ предоставляет удобные классы и методы для создания и манипулирования графическими объектами, такими как линии, прямоугольники, эллипсы, изображения и т.д. [2]

GDI+ поддерживает рендеринг 2D-графики, а также предоставляет возможности для работы с текстом, путями, растровыми и векторными изображениями, прозрачностью, анимацией и другими графическими эффектами.

Основные возможности GDI+ включают создание изображений, рисование на поверхностях (контекстах устройства), масштабирование, поворот, наложение графических объектов, работу с цветами и палитрами, обработку событий мыши и клавиатуры, а также сохранение и загрузку графических данных в различных форматах.

GDI+ является более новым и мощным инструментом для работы с графикой, чем GDI, и предоставляет разработчикам больше возможностей для создания визуально привлекательных приложений под Windows.

Вместе WinAPI и GDI+ обеспечивают разработчикам мощный инструментарий для создания приложений с графическим интерфейсом пользователей, обработки сообщений, работы с графикой и другими аспектами, связанными с операционной системой Windows.

3 ОПИСАНИЕ ФУНКЦИЙ ПРОГРАММЫ

Согласно формулировке задачи, были спроектированы следующие функции программы:

– изменение цвета текста и заднего фона приложения;

– изменение шрифта текста.

*Изменение цвета текста и заднего фона приложения:*

Для изменения цвета заднего фона и цвета текста требуется выбрать пункт “Background” в меню “Styles”. Цвета заднего фона изменяются в произвольном порядке, а цвет текста меняется на цвет, противоположный цвету заднего фона (см. рисунок 1).



Рисунок 1 – Графический интерфейс программы

*Изменение шрифта текста:*

Для изменения шрифта текста в программе пользователю требуется в меню “Styles” кликнуть на пункт “Text”, а после этого выбрать один из трех стилей шрифта (см. рисунок 2): Times New Roman, Arials и Calibri. После этих действий один из данных шрифтов применится к надписи “My Minesweeper” (см. рисунок 3).

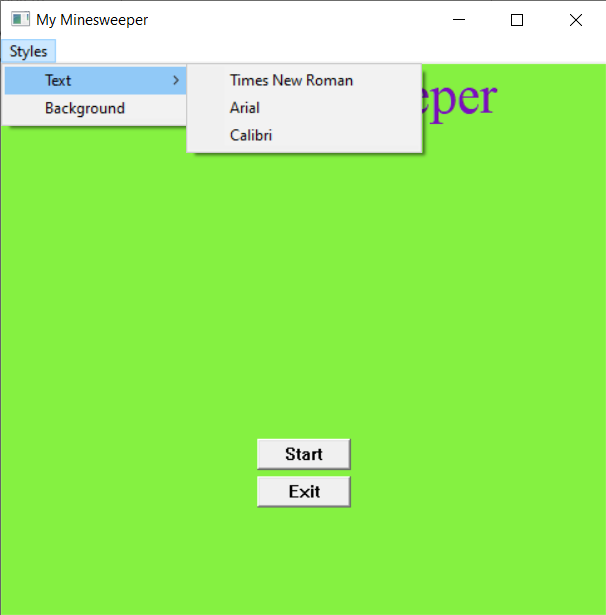


Рисунок 2 – Графический интерфейс вкладки меню



Рисунок 2 – Надпись “My Minesweeper” со шрифтом Calibri

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] КАК рисовать в Win32 API? [Электронный ресурс]. – Электронные данные. – Режим доступа: https://radiofront.narod.ru/htm/prog/htm/winda/api/paint.html#s

[3] GDI+ - Win32 Apps [Электронный ресурс]. – Электронные данные. – Режим доступа: https://learn.microsoft.com/ru-ru/windows/win32/gdiplus/-gdiplus-gdi-start

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

#define TEXT\_MENU1 1111 // меню стилей текста

#define TEXT\_MENU2 1112

#define TEXT\_MENU3 1113

#define BG\_MENU 2222 // цвет заднего фона

typedef struct { // клетка поля

bool isMine;

bool isFlag;

bool isOpen;

int minesAround;

} FCell;

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

int closedCells; // закрытые ячейки

RECT winRectangle;

HBRUSH brushRect = CreateSolidBrush(RGB(255, 255, 255));

HFONT fontText;

COLORREF fontColor;

GdiplusStartupInput gdiplusstartupInput;

void new\_game(HWND hwnd);

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

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

//void OnPaint(HDC hdc);

void MainWndAddMenus(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

{

fontText = CreateFontA(50, 0, 0, 0, FW\_NORMAL, FALSE, FALSE, FALSE,

DEFAULT\_CHARSET, OUT\_DEFAULT\_PRECIS, CLIP\_DEFAULT\_PRECIS,

DEFAULT\_QUALITY, DEFAULT\_PITCH | FF\_DONTCARE, "Times New Roman");

// 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;

ULONG\_PTR gdiplustokend;

GdiplusStartup(&gdiplustokend, &gdiplusstartupInput, NULL);

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 };

return 0;

case WM\_DESTROY:

PostQuitMessage(0);

return 0;

case WM\_PAINT:

{

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hwnd, &ps);

Graphics graphics(hdc);

FillRect(ps.hdc, &winRectangle, brushRect);

SetBkMode(ps.hdc, TRANSPARENT);

SetTextColor(ps.hdc, fontColor);

SelectObject(ps.hdc, fontText);

DrawTextA(ps.hdc, "My Minesweeper", -1, &winRectangle, DT\_SINGLELINE | DT\_CENTER | DT\_NOCLIP);

EndPaint(hwnd, &ps);

}

return 0;

case WM\_COMMAND:

{

if (wParam == TEXT\_MENU1) {

fontText = CreateFontA(50, 0, 0, 0, FW\_NORMAL, FALSE, FALSE, FALSE,

DEFAULT\_CHARSET, OUT\_DEFAULT\_PRECIS, CLIP\_DEFAULT\_PRECIS,

DEFAULT\_QUALITY, DEFAULT\_PITCH | FF\_DONTCARE, "Times New Roman");

RedrawWindow(hwnd, NULL, NULL, RDW\_UPDATENOW | RDW\_INVALIDATE);

return 0;

}

else if (wParam == TEXT\_MENU2) {

fontText = CreateFontA(50, 0, 0, 0, FW\_NORMAL, FALSE, FALSE, FALSE,

DEFAULT\_CHARSET, OUT\_DEFAULT\_PRECIS, CLIP\_DEFAULT\_PRECIS,

DEFAULT\_QUALITY, DEFAULT\_PITCH | FF\_DONTCARE, "Arial");

RedrawWindow(hwnd, NULL, NULL, RDW\_UPDATENOW | RDW\_INVALIDATE);

return 0;

}

else if (wParam == TEXT\_MENU3) {

fontText = CreateFontA(50, 0, 0, 0, FW\_NORMAL, FALSE, FALSE, FALSE,

DEFAULT\_CHARSET, OUT\_DEFAULT\_PRECIS, CLIP\_DEFAULT\_PRECIS,

DEFAULT\_QUALITY, DEFAULT\_PITCH | FF\_DONTCARE, "Calibri");

RedrawWindow(hwnd, NULL, NULL, RDW\_UPDATENOW | RDW\_INVALIDATE);

return 0;

}

else if (wParam == BG\_MENU) {

srand(time(NULL));

int colorR = rand() % 255;

int colorG = rand() % 255;

int colorB = rand() % 255;

brushRect = CreateSolidBrush(

RGB(colorR,

colorG,

colorB

));

fontColor = RGB(255 - colorR, 255 - colorG, 255 - colorB);

RedrawWindow(hwnd, NULL, NULL, RDW\_UPDATENOW | RDW\_INVALIDATE);

return 0;

}

else 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);

}

}

}

}

void MainWndAddMenus(HWND hWnd) {

HMENU RootMenu = CreateMenu();

HMENU SubMenu = CreateMenu();

HMENU SubFontMenu = CreateMenu();

AppendMenu(SubFontMenu, MF\_STRING, TEXT\_MENU1, L"Times New Roman");

AppendMenu(SubFontMenu, MF\_STRING, TEXT\_MENU2, L"Arial");

AppendMenu(SubFontMenu, MF\_STRING, TEXT\_MENU3, L"Calibri");

AppendMenu(SubMenu, MF\_POPUP, (UINT\_PTR)SubFontMenu, L"Text");

AppendMenu(SubMenu, MF\_STRING, BG\_MENU, L"Background");

AppendMenu(RootMenu, MF\_POPUP, (UINT\_PTR)SubMenu, L"Styles");

SetMenu(hWnd, RootMenu);

}