*Министерство образования Республики Беларусь*

*Учреждение образования*

*«Брестский государственный технический университет»*

*Кафедра ИИТ*

Лабораторная работа №2

По дисциплине: «Современные средства вычислительной техники»

Тема:«**Введение в DirectDraw**»

Выполнил:

Студент 3-го курса

группы ИИ-23

Романюк А.П.

Проверил:

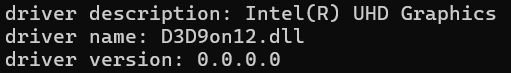
Кулеша В. И.

Брест 2025

**Задание 1**

**Цель:** Изучить основные теоретические сведения о DirectDraw. Разработать программу получения информации о драйвере видеопроцессора.

**Задание:** Необходимо написать консольную программу, которая инициализирует DirectDraw и получает информацию об используемом драйвере видеопроцессора.

**Ход работы**

#include <windows.h>

#include <ddraw.h>

#include <iostream>

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

// Вручную определяем IID\_IDirectDraw7, если он не определен в заголовочных файлах

const IID IID\_IDirectDraw7 = { 0x15e65ec0, 0x3b9c, 0x11d2, {0xb9, 0x2f, 0x00, 0x60, 0x97, 0x97, 0xea, 0x5b} };

int main() {

HRESULT hr;

LPDIRECTDRAW7 pDD = nullptr;

LPDIRECTDRAWENUMERATEEXA pDDEnum = nullptr;

// Инициализация DirectDraw

hr = DirectDrawCreateEx(nullptr, (LPVOID\*)&pDD, IID\_IDirectDraw7, nullptr);

if (FAILED(hr)) {

std::cerr << "Ошибка инициализации DirectDraw: " << hr << std::endl;

return 1;

}

// Получение информации о драйвере

DDDEVICEIDENTIFIER2 ddIdentifier;

hr = pDD->GetDeviceIdentifier(&ddIdentifier, 0);

if (FAILED(hr)) {

std::cerr << "Ошибка получения информации о драйвере: " << hr << std::endl;

pDD->Release();

return 1;

}

// Вывод информации о драйвере

std::cout << "driver description: " << ddIdentifier.szDescription << std::endl;

std::cout << "driver name: " << ddIdentifier.szDriver << std::endl;

std::cout << "driver version: " << HIWORD(ddIdentifier.liDriverVersion.HighPart) << "."

<< LOWORD(ddIdentifier.liDriverVersion.HighPart) << "."

<< HIWORD(ddIdentifier.liDriverVersion.LowPart) << "."

<< LOWORD(ddIdentifier.liDriverVersion.LowPart) << std::endl;

// Освобождение ресурсов

pDD->Release();

return 0;

}

**Задание 2**

**Цель:** Изучить базовые принципы программирования DirectDraw. Разработать программу, применяя полученные знания.

**Задание:** Необходимо написать программу, которая отображает на экране "звездное небо" - набор пикселов разных цветов.

#include <windows.h>

#include <ddraw.h>

#include <vector>

#include <cstdlib>

#include <ctime>

#include <tchar.h>

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

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

const int WIDTH = 800;

const int HEIGHT = 600;

const int STARS = 1000;

const int STAR\_LIFETIME = 100;

struct Star {

int x, y;

COLORREF color;

int lifetime;

};

LPDIRECTDRAW7 pDD = nullptr;

LPDIRECTDRAWSURFACE7 pPrimary = nullptr;

LPDIRECTDRAWSURFACE7 pBackBuffer = nullptr;

std::vector<Star> stars;

COLORREF RandomColor() {

return RGB(rand() % 256, rand() % 256, rand() % 256);

}

bool InitDirectDraw(HWND hwnd) {

HRESULT hr;

hr = DirectDrawCreateEx(nullptr, (LPVOID\*)&pDD, IID\_IDirectDraw7, nullptr);

if (FAILED(hr)) {

MessageBox(hwnd, TEXT("Ошибка создания DirectDraw!"), TEXT("Ошибка"), MB\_ICONERROR);

return false;

}

hr = pDD->SetCooperativeLevel(hwnd, DDSCL\_EXCLUSIVE | DDSCL\_FULLSCREEN);

if (FAILED(hr)) {

MessageBox(hwnd, TEXT("Ошибка установки режима DirectDraw!"), TEXT("Ошибка"), MB\_ICONERROR);

return false;

}

hr = pDD->SetDisplayMode(WIDTH, HEIGHT, 32, 0, 0);

if (FAILED(hr)) {

MessageBox(hwnd, TEXT("Ошибка установки разрешения!"), TEXT("Ошибка"), MB\_ICONERROR);

return false;

}

DDSURFACEDESC2 ddsd = {};

ddsd.dwSize = sizeof(ddsd);

ddsd.dwFlags = DDSD\_CAPS | DDSD\_BACKBUFFERCOUNT;

ddsd.ddsCaps.dwCaps = DDSCAPS\_PRIMARYSURFACE | DDSCAPS\_FLIP | DDSCAPS\_COMPLEX;

ddsd.dwBackBufferCount = 1;

hr = pDD->CreateSurface(&ddsd, &pPrimary, nullptr);

if (FAILED(hr)) {

MessageBox(hwnd, TEXT("Ошибка создания первичной поверхности!"), TEXT("Ошибка"), MB\_ICONERROR);

return false;

}

DDSCAPS2 ddscaps = {};

ddscaps.dwCaps = DDSCAPS\_BACKBUFFER;

hr = pPrimary->GetAttachedSurface(&ddscaps, &pBackBuffer);

if (FAILED(hr)) {

MessageBox(hwnd, TEXT("Ошибка получения заднего буфера!"), TEXT("Ошибка"), MB\_ICONERROR);

return false;

}

return true;

}

void DrawStars() {

DDSURFACEDESC2 ddsd = {};

ddsd.dwSize = sizeof(ddsd);

pBackBuffer->Lock(nullptr, &ddsd, DDLOCK\_WAIT, nullptr);

BYTE\* buffer = (BYTE\*)ddsd.lpSurface;

int pitch = ddsd.lPitch;

memset(buffer, 0, pitch \* HEIGHT);

for (const auto& star : stars) {

BYTE\* pixel = buffer + star.y \* pitch + star.x \* 4;

\*(COLORREF\*)pixel = star.color;

}

pBackBuffer->Unlock(nullptr);

}

void UpdateStars() {

for (auto& star : stars) {

star.lifetime--;

if (star.lifetime <= 0) {

star.x = rand() % WIDTH;

star.y = rand() % HEIGHT;

star.color = RandomColor();

star.lifetime = STAR\_LIFETIME;

}

}

}

void Render(HWND hwnd) {

if (!pPrimary || !pBackBuffer) return;

UpdateStars();

DrawStars();

pPrimary->Flip(nullptr, DDFLIP\_WAIT);

}

void Cleanup() {

if (pBackBuffer) pBackBuffer->Release();

if (pPrimary) pPrimary->Release();

if (pDD) pDD->Release();

}

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

switch (uMsg) {

case WM\_CREATE:

srand(static\_cast<unsigned int>(time(nullptr)));

for (int i = 0; i < STARS; ++i) {

stars.push\_back({ rand() % WIDTH, rand() % HEIGHT, RandomColor(), rand() % STAR\_LIFETIME });

}

if (!InitDirectDraw(hwnd)) {

PostQuitMessage(0);

}

SetTimer(hwnd, 1, 16, nullptr);

break;

case WM\_TIMER:

Render(hwnd);

break;

case WM\_DESTROY:

KillTimer(hwnd, 1);

Cleanup();

PostQuitMessage(0);

break;

default:

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

}

return 0;

}

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow) {

const TCHAR\* CLASS\_NAME = TEXT("StarWindowClass");

WNDCLASS wc = {};

wc.lpfnWndProc = WindowProc;

wc.hInstance = hInstance;

wc.lpszClassName = CLASS\_NAME;

RegisterClass(&wc);

HWND hwnd = CreateWindowEx(

0, CLASS\_NAME, TEXT("Звездное небо (DirectDraw)"), WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, CW\_USEDEFAULT, WIDTH, HEIGHT,

nullptr, nullptr, hInstance, nullptr

);

if (hwnd == nullptr) {

return 0;

}

ShowWindow(hwnd, SW\_MAXIMIZE);

MSG msg = {};

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

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return 0;

}

**Вывод:** изучил базовые принципы программирования DirectDraw. Разработал программы, применяя полученные знания.