Міністерство освіти і науки України Харківський національний університет радіоелектроніки

Кафедра Інформатики

Звіт

з Лабораторної роботи №5

з дисципліни

«Операційні системи»

Виконав: Перевірив:

ITIHФ-20-1 Професор

Самченко С.О. Сінельнікова Т.Ф.

Харків – 2022

**Dll**

Необхідно розробити програму client, яка дозволятиме відправляти команди на сервер по локальній мережі по протоколу TCP / IP. У сервері для обробки кожної команди створюється власний потік. Клієнтська програма повинна дозволяти відправити команду малювання кола, прямокутника, текстового повідомлення.Кожен об'єкт в клієнтському додатку задається набором параметрів, в тому числі колір і градієнтна заливка. Відправка команди виконується по натисканні на певну клавішу. Малювання об'єктів у серверному додатку виконується на певній робочій області розміром 640х480 пікселів. Повинні відбуватися перевірки коректності заданих параметрів в командах, щоб можна було забезпечити виведення об'єктів. Малювання об'єктів повинно виконуватися за допомогою GDI +. Дані про кількість намальованих об'єктів кожного типу необхідно зберегти в реєстрі. Після першого запуску перша програма повинна поміщати іконку в Windows Tray і при натисканні комбінації клавіш Ctrl + P має виводитися вікно з інформацією про кількість намальованих об'єктів кожного типу, а при натисканні на комбінацію клавіш Alt + T ці дані повинні видалятися з реєстру.

**Dll.framework.h:**

#pragma once

#define WIN32\_LEAN\_AND\_MEAN

#include <windows.h>

**Dll.pch.h:**

// pch.h: это предварительно скомпилированный заголовочный файл.

// Перечисленные ниже файлы компилируются только один раз, что ускоряет последующие сборки.

// Это также влияет на работу IntelliSense, включая многие функции просмотра и завершения кода.

// Однако изменение любого из приведенных здесь файлов между операциями сборки приведет к повторной компиляции всех(!) этих файлов.

// Не добавляйте сюда файлы, которые планируете часто изменять, так как в этом случае выигрыша в производительности не будет.

#ifndef PCH\_H

#define PCH\_H

// Добавьте сюда заголовочные файлы для предварительной компиляции

#include "framework.h"

#endif //PCH\_H

**Dll.** **Source.cpp:**

#include "pch.h"

#include <iostream>

#include <Windows.h>

struct Shape

{

struct Color {

BYTE r;

BYTE g;

BYTE b;

Color(BYTE \_r, BYTE \_g, BYTE \_b)

{

r = \_r;

g = \_g;

b = \_b;

}

};

int posX;

int posY;

int size;

Color\* color1;

Color\* color2;

Shape(int \_x, int \_y, int \_size)

{

posX = \_x;

posY = \_y;

size = \_size;

}

void SetColorStart(int red, int green, int blue)

{

color1 = new Color(red, green, blue);

}

void SetColorEnd(int red, int green, int blue)

{

color2 = new Color(red, green, blue);

}

};

extern "C" \_\_declspec(dllexport) Shape \_cdecl GenerateShape()

{

Shape shape(rand() % 640, rand() % 480, rand() % 100);

shape.SetColorStart(rand() % 255, rand() % 255, rand() % 255);

shape.SetColorEnd(rand() % 255, rand() % 255, rand() % 255);

return shape;

}

**Project1.** **Source.cpp**

#include <iostream>

#include <Windows.h>

#include <string>

using namespace std;

struct Shape

{

struct Color {

BYTE r;

BYTE g;

BYTE b;

};

int posX;

int posY;

int size;

Color\* color1;

Color\* color2;

string PrintColor()

{

string str;

str += to\_string((int)color1->r) + "," + to\_string((int)color1->g) + "," + to\_string((int)color1->b) + "; ";

str += to\_string((int)color2->r) + "," + to\_string((int)color2->g) + "," + to\_string((int)color2->b);

return str;

}

};

Shape(\_cdecl \*GenerateShape)();

int main()

{

HMODULE dll = LoadLibrary("Dll.dll");

if (FAILED(dll))

{

MessageBox(NULL,

("Невозможно загрузить библиотеку."),

"title",

MB\_ICONERROR | MB\_OK);

return 1;

}

GenerateShape = (Shape(\*)())

GetProcAddress(dll, "GenerateShape");

if (GenerateShape == NULL)

{

FreeLibrary(dll);

MessageBox(NULL,

("В библиотеке не найдена функция GenerateShape."),

"title",

MB\_ICONERROR | MB\_OK);

return 1;

}

Shape shape[5];

for (int i = 0; i < 5; ++i)

{

shape[i] = GenerateShape();

cout << "Shape:\n";

cout << " - Position: " << shape[i].posX << " " << shape[i].posY << endl;

cout << " - Size: " << shape[i].size << endl;

cout << " - Gradient: (" << shape[i].PrintColor() << ")" << endl;

cout << "--------------------------------" << endl;

}

system("pause");

return 0;

}