Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Операционные системы и системное программирование

Отчёт

к лабораторной работе №5

на тему

Реестр и журналы (Windows). Доступ к реестру Windows. Работа с

журналами Windows. Другие вспомогательные средства управления.

Студент: гр.153502

Кирзнер А. П.

Проверил: Гриценко Н.Ю.

Минск 2023

**СОДЕРЖАНИЕ**

1 Теоретические сведения 3

2 Результат выполнения программы 4

Список использованных источников 5

Приложение А 6

# 1 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Реестр – это определяемая системой база данных, в которой приложения и системные компоненты хранят и извлекают данные конфигурации. Данные, хранящиеся в реестре, зависят от версии *Microsoft Windows*. Приложения используют *API* реестра для получения, изменения или удаления данных реестра [1].

Функция *RegOpenKeyEx* открывает указанный ключ реестра (*HKEY\_CLASSES\_ROOT*, *HKEY\_CURRENT\_CONFIG*, *HKEY\_CURRENT\_USER*, *HKEY\_LOCAL\_MACHINE*, *HKEY\_USERS*) [2].

Функция *RegQueryInfoKey* получает информацию об указанном разделе реестра [3].

Функция *RegEnumKeyEx* перечисляет подразделы указанного открытого раздела реестра. Функция получает информацию об одном подразделе при каждом вызове. Чтобы перечислить подразделы, приложение должно сначала вызвать функцию *RegEnumKeyEx* с параметром *dwIndex*, установленным в ноль. Затем приложение должно увеличить параметр *dwIndex* и вызывать *RegEnumKeyEx* до тех пор, пока не исчезнут подразделы (это означает, что функция возвращает *ERROR\_NO\_MORE\_ITEMS*).

Приложение также может установить *dwIndex* в индекс последнего подраздела при первом вызове функции и уменьшать индекс до тех пор, пока не будет пронумерован подраздел с индексом 0. Чтобы получить индекс последнего подраздела, используйте функцию *RegQueryInfoKey* [4].

# 2 РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ ПРОГРАММЫ

Главное окно приложения состоит из текстового вывода путей, по которым расположены записи с не присвоенными значением, которые предполагаются возможно устаревшими или ненужными (рисунок 1).

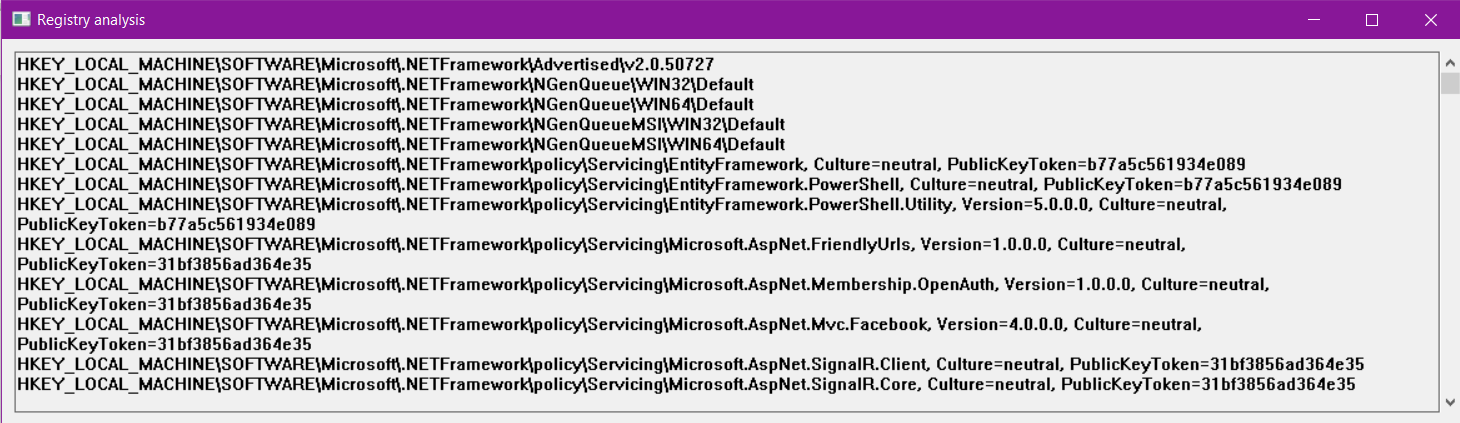


Рисунок 1 – Графический интерфейс программы

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Реестр [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/ru-ru/windows/win32/sysinfo/registry>.

[2] Функция RegOpenKeyExA (winreg.h) [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/en-us/windows/win32/api/winreg/nf-winreg-regopenkeyexa>.

[3] Функция RegQueryInfoKeyA (winreg.h) [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/en-us/windows/win32/api/winreg/nf-winreg-regqueryinfokeya>.

[4] Функция RegEnumKeyExA (winreg.h) [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/en-us/windows/win32/api/winreg/nf-winreg-regenumkeyexa>.

# ПРИЛОЖЕНИЕ А

**(обязательное)**

**Исходный код программы**

**Файл Lab5.h**

#pragma once

LRESULT CALLBACK SoftwareMainProcedure(HWND hWnd, UINT msg, WPARAM wp, LPARAM lp);

WNDCLASS NewWindowClass(HBRUSH BGColor, HCURSOR Cursor, HINSTANCE hInst, HICON Icon, LPCWSTR Name, WNDPROC Procedure);

HWND hEdit;

HANDLE hFile;

void EnumerateRegistryKeys(HKEY hKey, const std::wstring& path, HWND hEdit);

**Файл Lab5.cpp**

#include <Windows.h>

#include <thread>

#include <string>

#include "Lab5.h"

int WINAPI WinMain(HINSTANCE hInst, HINSTANCE hPrevInst, LPSTR args, int ncmdshows) {

WNDCLASS SoftwareMainClass = NewWindowClass((HBRUSH)COLOR\_WINDOW, LoadCursor(NULL, IDC\_ARROW), hInst, LoadIcon(NULL, IDI\_APPLICATION), L"MainWndClass", SoftwareMainProcedure);

if (!RegisterClass(&SoftwareMainClass)) { return -1; }

MSG SoftwareMainMessage = { 0 };

HWND hWnd = CreateWindow(L"MainWndClass", L"Registry analysis", WS\_OVERLAPPEDWINDOW | WS\_VISIBLE, 100, 100, 500, 250, NULL, NULL, NULL, NULL);

ShowWindow(hWnd, ncmdshows);

UpdateWindow(hWnd);

while (GetMessage(&SoftwareMainMessage, NULL, NULL, NULL)) {

TranslateMessage(&SoftwareMainMessage);

DispatchMessage(&SoftwareMainMessage);

}

return 0;

}

WNDCLASS NewWindowClass(HBRUSH BGColor, HCURSOR Cursor, HINSTANCE hInst, HICON Icon, LPCWSTR Name, WNDPROC Procedure) {

WNDCLASS NWC = { 0 };

NWC.hCursor = Cursor;

NWC.hIcon = Icon;

NWC.hInstance = hInst;

NWC.lpszClassName = Name;

NWC.hbrBackground = BGColor;

NWC.lpfnWndProc = Procedure;

return NWC;

}

LRESULT CALLBACK SoftwareMainProcedure(HWND hWnd, UINT msg, WPARAM wp, LPARAM lp) {

switch (msg) {

case WM\_CREATE: {

RECT clientRect;

GetClientRect(hWnd, &clientRect);

hEdit = CreateWindowW(

L"EDIT",

L"",

WS\_VISIBLE | WS\_CHILD | WS\_BORDER | ES\_AUTOVSCROLL | ES\_READONLY | ES\_MULTILINE | WS\_VSCROLL,

10,

10,

(clientRect.right - 10),

clientRect.bottom - 20,

hWnd,

NULL,

(HINSTANCE)GetWindowLongPtr(hWnd, GWLP\_HINSTANCE),

NULL

);

HKEY hKey;

// HKEY\_CLASSES\_ROOT

// HKEY\_CURRENT\_USER

// HKEY\_LOCAL\_MACHINE

// HKEY\_USERS

// HKEY\_CURRENT\_CONFIG

if (RegOpenKeyEx(HKEY\_LOCAL\_MACHINE, L"SOFTWARE\\Microsoft", 0, KEY\_READ, &hKey) == ERROR\_SUCCESS) {

EnumerateRegistryKeys(hKey, L"HKEY\_LOCAL\_MACHINE\\SOFTWARE\\Microsoft", hEdit);

RegCloseKey(hKey);

}

break;

}

case WM\_SIZE:

RECT clientRect;

GetClientRect(hWnd, &clientRect);

MoveWindow(

hEdit,

10,

10,

(clientRect.right - 10),

clientRect.bottom - 20,

TRUE);

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

default: return DefWindowProc(hWnd, msg, wp, lp);

}

}

void EnumerateRegistryKeys(HKEY hKey, const std::wstring& path, HWND hEdit) {

DWORD dwBytesWritten;

DWORD subKeys;

DWORD maxSubKeyLen;

DWORD values;

DWORD maxValueNameLen;

DWORD maxValueDataLen;

if (RegQueryInfoKey(hKey, NULL, NULL, NULL, &subKeys, &maxSubKeyLen, NULL, &values, &maxValueNameLen, &maxValueDataLen, NULL, NULL) != ERROR\_SUCCESS) {

return;

}

if (values == 0 && subKeys == 0) {

SendMessageW(hEdit, EM\_SETSEL, -1, -1);

SendMessageW(hEdit, EM\_REPLACESEL, 0, (LPARAM)(path + L"\r\n").c\_str());

return;

}

for (DWORD i = 0; i < subKeys; ++i) {

WCHAR subKeyName[256];

DWORD subKeyNameSize = sizeof(subKeyName);

if (RegEnumKeyEx(hKey, i, subKeyName, &subKeyNameSize, NULL, NULL, NULL, NULL) == ERROR\_SUCCESS) {

HKEY subKey;

if (RegOpenKeyEx(hKey, subKeyName, 0, KEY\_READ, &subKey) == ERROR\_SUCCESS) {

EnumerateRegistryKeys(subKey, path + L"\\" + subKeyName, hEdit);

RegCloseKey(subKey);

}