Одесский национальный политехнический университет

Кафедра интеллектуальных систем и сетей

Лабораторная работа №6

По предмету «Системное программирование»

На тему «Динамически подключаемые библиотеки»

Выполнили

Студенты группы АЕ-145

Михайлов О.М.

Зайцева М.О.

Проверил

Олещук О.В.

Одессы 2017

// Lab6.cpp: определяет точку входа для приложения.

//

#include "stdafx.h"

#include "Laba6.h"

typedef int(WINAPI\* Add) (int, int);

typedef BOOL(WINAPI\* Message1) (HWND, LPCWSTR);

#define MAX\_LOADSTRING 100

// Глобальные переменные:

HINSTANCE hInst; // текущий экземпляр

WCHAR szTitle[MAX\_LOADSTRING]; // Текст строки заголовка

WCHAR szWindowClass[MAX\_LOADSTRING]; // имя класса главного окна

static HINSTANCE hLibrary;

static Add addFunc;

static Message1 messageFunc;

int a = 1;

int b = 4;

bool trigger = false;

// Отправить объявления функций, включенных в этот модуль кода:

ATOM MyRegisterClass(HINSTANCE hInstance);

BOOL InitInstance(HINSTANCE, int);

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

INT\_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);

int APIENTRY wWinMain(\_In\_ HINSTANCE hInstance,

\_In\_opt\_ HINSTANCE hPrevInstance,

\_In\_ LPWSTR lpCmdLine,

\_In\_ int nCmdShow)

{

UNREFERENCED\_PARAMETER(hPrevInstance);

UNREFERENCED\_PARAMETER(lpCmdLine);

// TODO: разместите код здесь.

// Инициализация глобальных строк

LoadStringW(hInstance, IDS\_APP\_TITLE, szTitle, MAX\_LOADSTRING);

LoadStringW(hInstance, IDC\_LABA6, szWindowClass, MAX\_LOADSTRING);

MyRegisterClass(hInstance);

// Выполнить инициализацию приложения:

if (!InitInstance(hInstance, nCmdShow))

{

return FALSE;

}

HACCEL hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC\_LABA6));

MSG msg;

// Цикл основного сообщения:

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

{

if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

}

return (int)msg.wParam;

}

//

// ФУНКЦИЯ: MyRegisterClass()

//

// НАЗНАЧЕНИЕ: регистрирует класс окна.

//

ATOM MyRegisterClass(HINSTANCE hInstance)

{

WNDCLASSEXW wcex;

wcex.cbSize = sizeof(WNDCLASSEX);

wcex.style = CS\_HREDRAW | CS\_VREDRAW;

wcex.lpfnWndProc = WndProc;

wcex.cbClsExtra = 0;

wcex.cbWndExtra = 0;

wcex.hInstance = hInstance;

wcex.hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(IDI\_LABA6));

wcex.hCursor = LoadCursor(nullptr, IDC\_ARROW);

wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);

wcex.lpszMenuName = MAKEINTRESOURCEW(IDC\_LABA6);

wcex.lpszClassName = szWindowClass;

wcex.hIconSm = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI\_SMALL));

return RegisterClassExW(&wcex);

}

//

// ФУНКЦИЯ: InitInstance(HINSTANCE, int)

//

// НАЗНАЧЕНИЕ: сохраняет обработку экземпляра и создает главное окно.

//

// КОММЕНТАРИИ:

//

// В данной функции дескриптор экземпляра сохраняется в глобальной переменной, а также

// создается и выводится на экран главное окно программы.

//

BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)

{

hInst = hInstance; // Сохранить дескриптор экземпляра в глобальной переменной

HWND hWnd = CreateWindowW(szWindowClass, szTitle, WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, 0, CW\_USEDEFAULT, 0, nullptr, nullptr, hInstance, nullptr);

if (!hWnd)

{

return FALSE;

}

ShowWindow(hWnd, nCmdShow);

UpdateWindow(hWnd);

return TRUE;

}

//

// ФУНКЦИЯ: WndProc(HWND, UINT, WPARAM, LPARAM)

//

// НАЗНАЧЕНИЕ: обрабатывает сообщения в главном окне.

//

// WM\_COMMAND — обработать меню приложения

// WM\_PAINT — отрисовать главное окно

// WM\_DESTROY — отправить сообщение о выходе и вернуться

//

//

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

{

LPCWSTR \*name = new LPCWSTR[5];

name[0] = \_T("DLLNumber1");

name[1] = \_T("DLLNumber2");

name[2] = \_T("DLLNumber3");

switch (message)

{

case WM\_COMMAND:

{

int wmId = LOWORD(wParam);

// Разобрать выбор в меню:

switch (wmId)

{

case IDM\_ABOUT:

DialogBox(hInst, MAKEINTRESOURCE(IDD\_ABOUTBOX), hWnd, About);

break;

case IDM\_EXIT:

DestroyWindow(hWnd);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

}

break;

case WM\_CHAR:

{

switch (wParam)//тут нажатиями на клавиатуре можно уменьшать число, которые складываться будут

{

case 61://знак минус меняет левое число

trigger ? a++ : a--; //тернарная операция, которая зависит от состояния trigger

break;

case 45://знак равно меняет правое число

trigger ? b++ : b--;

break;

case 32: //пробелом меняем направление сложения

trigger ? trigger = false : trigger = true;

break;

default:

break;

}

RECT r;

GetClientRect(hWnd, &r);

InvalidateRect(hWnd, &r, TRUE);

break;

}

case WM\_CREATE:

hLibrary = LoadLibrary(\_T("AddDLLMy"));//подключаем библиотеку, которая содержит функцию Add

if (hLibrary)

{

addFunc = (Add)GetProcAddress(hLibrary, "Add");//получаем функцию для сложения из библиотеки

if (addFunc == NULL) {

MessageBox(hWnd, \_T("Функция Add не найдена"), \_T("LoadLibrary"), MB\_OK | MB\_ICONQUESTION);

DestroyWindow(hWnd);

return 0;

}

messageFunc = (Message1)GetProcAddress(hLibrary, "Message");

if (addFunc == NULL) {

MessageBox(hWnd, \_T("Функция Message не найдена"), \_T("LoadLibrary"), MB\_OK | MB\_ICONQUESTION);

DestroyWindow(hWnd);

return 0;

}

}

else {

MessageBox(hWnd, \_T("Библиотека не найдена"), \_T("LoadLibrary"), MB\_OK | MB\_ICONQUESTION);

DestroyWindow(hWnd);

return 0;

}

break;

case WM\_PAINT:

{

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hWnd, &ps);

char buf[100] = { '\0' };

char text[100] = { '\0' };

int StringCounter = 0;

\_itoa\_s(a, buf, 10);//вывод первого числа

strcat\_s(text, buf);

strcat\_s(text, " + ");

\_itoa\_s(b, buf, 10);//вывод второго числа

strcat\_s(text, buf);

strcat\_s(text, " = ");

\_itoa\_s(addFunc(a, b), buf, 10);//вывод на экран сложения в результате операции Add

strcat\_s(text, buf);

TextOutA(hdc, 0, (StringCounter++) \* 20, (LPCSTR)text, strlen(text));//вывод выше полученных результатов

HINSTANCE library;

int libraryCounter = 0;

TextOutW(hdc, 0, (StringCounter++) \* 20, L"Найдены следующие номера DLL-библеотек с функцией \"HelloWorld\":", 64);

while (libraryCounter<3) {

library = LoadLibrary(name[libraryCounter]); //подключаем библиотеку, которая надо проверить на содержание функции HelloWorld

if (library == NULL) break;

if (GetProcAddress(library, "HelloWorld") != NULL) { //запрашиваем функцию HelloWorld и если она есть, то выводим имя

TextOutA(hdc, 0, (StringCounter++) \* 20, (LPCSTR)(name[libraryCounter]),20); //вывод названий библиотек

FreeLibrary(library); //уменьшает итоговое число ссылок на загруженные динамически подключаемые библиотеки

}

libraryCounter++;

}

EndPaint(hWnd, &ps);

}

break;

case WM\_LBUTTONDOWN:

messageFunc(hWnd, \_T("HelloWorld!"));

break; case WM\_DESTROY:

FreeLibrary(hLibrary);

PostQuitMessage(0);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

return 0;

}

// Обработчик сообщений для окна "О программе".

INT\_PTR CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

switch (message)

{

case WM\_INITDIALOG:

return (INT\_PTR)TRUE;

case WM\_COMMAND:

if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)

{

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

}

break;

}

return (INT\_PTR)FALSE;

}

Библиотеки DLL:

1. Библиотека AddDLL:

// dllmain.cpp: определяет точку входа для приложения DLL.

#include "stdafx.h"

\_\_declspec(dllexport) int WINAPI Add(int, int);

\_\_declspec(dllexport) BOOL WINAPI Message(HWND, LPCWSTR);

BOOL APIENTRY DllMain(HMODULE hModule,

DWORD ul\_reason\_for\_call,

LPVOID lpReserved

)

{

switch (ul\_reason\_for\_call)

{

case DLL\_PROCESS\_ATTACH:

case DLL\_THREAD\_ATTACH:

case DLL\_THREAD\_DETACH:

case DLL\_PROCESS\_DETACH:

break;

}

return TRUE;

}

int WINAPI Add(int s1, int s2)

{

return s1 + s2;

}

BOOL WINAPI Message(HWND hWnd, LPCWSTR message)

{

MessageBox(hWnd, message, L"Внимание", 0);

return true;

}

2. Библиотека DLLNumber:

// dllmain.cpp: определяет точку входа для приложения DLL.

#include "stdafx.h"

\_\_declspec(dllexport) BOOL WINAPI HelloWorld(HWND);

BOOL APIENTRY DllMain(HMODULE hModule,

DWORD ul\_reason\_for\_call,

LPVOID lpReserved

)

{

switch (ul\_reason\_for\_call)

{

case DLL\_PROCESS\_ATTACH:

case DLL\_THREAD\_ATTACH:

case DLL\_THREAD\_DETACH:

case DLL\_PROCESS\_DETACH:

break;

}

return TRUE;

}

BOOL WINAPI HelloWorld(HWND hWnd)

{

MessageBox(hWnd, L"HelloWorld!", L"Внимание", 0);

return true;

}

**Вывод:** DLL-библеотеки - хорошие наборы подключаемых функций, вполне неплохая функция Windows, вот только подключаются они сложнее заголовков, но при этом не обязательно должны быть только в папке с программой, что впрочем не сильно существенно.