|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | | | |
| Институт информационных технологий (ИТ) | |
| Кафедра инструментального и прикладного программного обеспечения (ИиППО) | |

|  |  |
| --- | --- |
| **ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №3** | |
| **по дисциплине** | |
| «Системное программное обеспечение»  по теме: «Диалоговые окна» | |
|  | |
| Выполнил студент группы ИКБО-16-18 | Шамян Г.Г. |
| Принял преподаватель | Волков М.Ю. |

|  |  |  |
| --- | --- | --- |
| Работа выполнена | «\_\_»\_\_\_\_\_\_\_2021 г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_2021 г. | *(подпись руководителя)* |

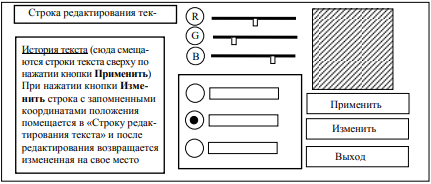
Москва 2021

**Цель работы**

Изучить вопросы проектирования и создания модальных и немодальных диалоговых панелей на базе Win32/64 API

**Задание**

Создать приложение (рис. 1), использующее модальный, системный модальный, немодальный и стандартный диалоги и элементы управления типа кнопка, меню, список, движок, поле для ввода.



*Рис. 1 – Пример диалогового окна*

**Ход работы**

Итоговая программа представлена в листинге 1. Результат работы представлен на рисунках 2-3.

Листинг 1

#include <Windows.h>

#include <tchar.h>

#include <stdlib.h>

#include <string.h>

#include <iostream>

HINSTANCE hInst;

HWND hWndScrollR, hWndScrollG, hWndScrollB;

HWND hWndAccpetButton, hWndProsto;

WNDCLASSEX wcex, wcex2;

HWND hWnd, hWnd2;

TCHAR ChildName[] = \_T("ChildFrame");

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

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

// Текущая позиция полосы просмотра

int nPosition;

using namespace std;

int CALLBACK WinMain(

\_In\_ HINSTANCE hInstance,

\_In\_opt\_ HINSTANCE hPrevInstane,

\_In\_ LPSTR lpCmdLine,

\_In\_ int nCmdShow)

{

//window 1

wcex;

HBRUSH greenBrush = CreateSolidBrush(RGB(204, 255, 204));

ZeroMemory(&wcex, sizeof(WNDCLASSEX));

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, IDI\_APPLICATION);

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

wcex.hbrBackground = greenBrush;

wcex.lpszMenuName = NULL;

wcex.lpszClassName = L"window class1";

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

if (!RegisterClassEx(&wcex)) {

MessageBox(NULL,

\_T("Window class creation failed for window 1"),

\_T("Window Class Failed"),

MB\_ICONERROR);

return 1;

}

hWnd = CreateWindow(

wcex.lpszClassName,

L"Parent Window",

WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, CW\_USEDEFAULT,

500, 500,

NULL,

NULL,

hInstance,

NULL

);

if (!hWnd)

{

MessageBox(NULL,

\_T("Call to CreateWindow1 failed!"),

\_T("Window Creation Filed"),

MB\_ICONERROR);

return 1;

}

UpdateWindow(hWnd);

HBRUSH brush = CreateSolidBrush(RGB(204, 200, 204));

ZeroMemory(&wcex2, sizeof(WNDCLASSEX));

wcex2.cbSize = sizeof(WNDCLASSEX);

wcex2.style = CS\_HREDRAW | CS\_VREDRAW;

wcex2.lpfnWndProc = ChildProc;

wcex2.cbClsExtra = 0;

wcex2.cbWndExtra = 0;

wcex2.hInstance = hInstance;

wcex2.hIcon = LoadIcon(hInstance, IDI\_APPLICATION);

wcex2.hCursor = LoadCursor(NULL, IDC\_ARROW);

wcex2.hbrBackground = greenBrush;

wcex2.lpszMenuName = NULL;

wcex2.lpszClassName = L"window class2";

wcex2.hIconSm = LoadIcon(wcex2.hInstance, IDI\_APPLICATION);

if (!RegisterClassEx(&wcex2)) {

MessageBox(NULL,

\_T("Window class creation failed for window 2"),

\_T("Window Class Failed"),

MB\_ICONERROR);

return 1;

}

hWnd2 = CreateWindow(

wcex2.lpszClassName,

L"Parent Window",

WS\_THICKFRAME | WS\_CHILD,

250, 0,

200, 200,

hWnd,

NULL,

hInstance,

NULL

);

if (!hWnd2)

{

MessageBox(NULL,

\_T("Call to CreateWindow1 failed!"),

\_T("Window Creation Filed"),

MB\_ICONERROR);

return 1;

}

UpdateWindow(hWnd2);

hWndScrollR = CreateWindow(\_T("scrollbar"), NULL, WS\_CHILD | WS\_VISIBLE,

10, 10, 200, 20, hWnd, (HMENU)1, hInst, NULL);

SetScrollRange(hWndScrollR, SB\_CTL, 0, 255, TRUE);

hWndScrollG = CreateWindow(\_T("scrollbar"), NULL, WS\_CHILD | WS\_VISIBLE,

10, 50, 200, 20, hWnd, (HMENU)2, hInst, NULL);

SetScrollRange(hWndScrollG, SB\_CTL, 0, 255, TRUE);

hWndScrollB = CreateWindow(\_T("scrollbar"), NULL, WS\_CHILD | WS\_VISIBLE,

10, 90, 200, 20, hWnd, (HMENU)3, hInst, NULL);

SetScrollRange(hWndScrollB, SB\_CTL, 0, 255, TRUE);

hWndAccpetButton = CreateWindow(\_T("BUTTON"), \_T("ПРИМЕНИТЬ"), WS\_CHILD | WS\_VISIBLE,

10, 150, 100, 20, hWnd, (HMENU)1001, hInst, NULL);

SetParent(hWnd2, hWnd);

ShowWindow(hWnd,

nCmdShow);

ShowWindow(hWnd2,

nCmdShow);

MSG msg;

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

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return (int)msg.wParam;

};

int i;

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

{

PAINTSTRUCT ps;

HDC hdc;

switch (message)

{

case WM\_PAINT:

hdc = BeginPaint(hWnd, &ps);

TextOut(hdc,

5, 5,

NULL, NULL);

EndPaint(hWnd, &ps);

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

case WM\_HSCROLL:

{

if (hWndScrollR == (HWND)lParam) i = 1;

if (hWndScrollG == (HWND)lParam) i = 2;

if (hWndScrollB == (HWND)lParam) i = 3;

// В зависимости от параметра сообщения

// изменяем текущую позицию

switch (wParam)

{

case SB\_PAGEDOWN:

{

nPosition += 10;

break;

}

case SB\_LINEDOWN:

{

nPosition += 1;

break;

}

case SB\_PAGEUP:

{

nPosition -= 10;

break;

}

case SB\_LINEUP:

{

nPosition -= 1;

break;

}

case SB\_TOP:

{

nPosition = 0;

break;

}

case SB\_BOTTOM:

{

nPosition = 200;

break;

}

case SB\_THUMBPOSITION:

{

nPosition = LOWORD(lParam);

break;

}

case SB\_THUMBTRACK:

{

nPosition = LOWORD(lParam);

break;

}

default:

break;

}

// Ограничиваем пределы изменения текущей

// позиции значениями от 1 до 200

if (nPosition > 255) nPosition = 255;

if (nPosition < 0) nPosition = 0;

// Устанавливаем ползунок полосы просмотра

// в соответствии с новым значением

// текущей позиции

switch (i) {

case 1:

SetScrollPos(hWndScrollR, SB\_CTL, nPosition, TRUE);

break;

case 2:

SetScrollPos(hWndScrollG, SB\_CTL, nPosition, TRUE);

break;

case 3:

SetScrollPos(hWndScrollB, SB\_CTL, nPosition, TRUE);

break;

}

return 0;

}

case WM\_COMMAND:

{

switch(wParam) {

case 1001: {

int r = GetScrollPos(hWndScrollR, SB\_CTL);

int g = GetScrollPos(hWndScrollG, SB\_CTL);

int b = GetScrollPos(hWndScrollB, SB\_CTL);

HBRUSH brush = CreateSolidBrush(RGB(r, g, b));

SetClassLongPtr(hWnd2, GCL\_HBRBACKGROUND, (LONG)brush);

InvalidateRect(hWnd2, NULL, TRUE);

break;

}

default:

break;

}

break;

}

default:

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

break;

}

return 0;

};

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

{

PAINTSTRUCT ps;

HDC hdc;

switch (message)

{

case WM\_PAINT:

hdc = BeginPaint(hWnd, &ps);

TextOut(hdc,

5, 5,

NULL, NULL);

EndPaint(hWnd, &ps);

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

case WM\_COMMAND:

{

break;

}

default:

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

break;

}

return 0;

};

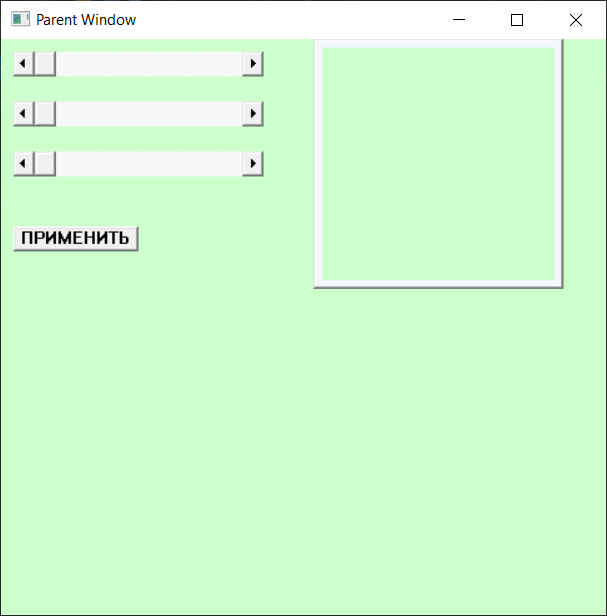


Рис.2 – Главное меню.

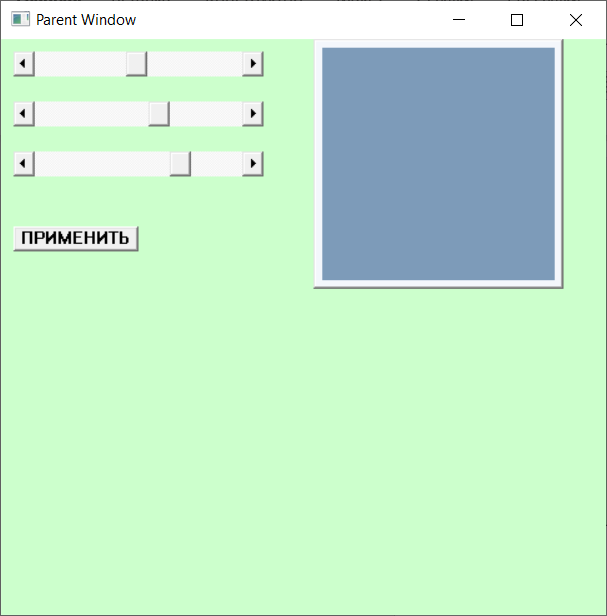


Рис.3 – После применения параметров цвета.

**Вывод**

В ходе практической работы была изучена работа с модальными и немодальными окнами с помощью WINAPI.

**Список использованных источников**

1. Пошаговое руководство. Создание классического оконного приложение [Электронный ресурс]. URL: https://docs.microsoft.com/ru-ru/cpp/windows/walkthrough-creating-windows-desktop-applications-cpp?view=msvc-160
2. Администрирование в Windows 10. [Электронный ресурс] : [https://docs.microsoft.com/ru-ru/windows/client-management/administrative-tools-in-windows-10 /](https://docs.microsoft.com/ru-ru/windows/client-management/administrative-tools-in-windows-10%20/) Ричард Ворд 2020 — 100 с.
3. «Тайм-менеджмент для системных администраторов». / Лимончелли Т.А. М.: Символ-плюс 2007 — 247 с. ISBN:978-5-93286-090-8