//------------------------------------------------------

// Файл: BMPVIEW.CPP

// Описание: Демонстрирует работу с растрами

//------------------------------------------------------

#define STRICT

#define WIN32\_LEAN\_AND\_MEAN

#include <windows.h>

#include <windowsx.h>

#include <stdlib.h>

#include <time.h>

#include <math.h>

#include <string.h>

#include "bmpview.h"

//Объекты контекста в памяти и шрифт

HDC hMemDC;

HBITMAP hMemBitmap;

HFONT hInfoFont;

//Два цвета для надписей

COLORREF crFontColor1;

COLORREF crFontColor2;

//Имя выводимого растра

char\* szFileName="sample.bmp";

//Ширина и высота экрана

UINT nScreenX;

UINT nScreenY;

//Реальные ширина и высота картинки

UINT nBmpWidth;

UINT nBmpHeight;

//Ширина и высота картинки при выводе

UINT nVBmpWidth;

UINT nVBmpHeight;

//Отступ сверху для надписи

UINT nTopDown=100;

int WINAPI WinMain(HINSTANCE hInst, HINSTANCE hPrevInst, LPSTR lpszCmdLine, int nCmdShow)

{

WNDCLASSEX wndClass;

HWND hWnd;

MSG msg;

nScreenX=GetSystemMetrics(SM\_CXSCREEN);

nScreenY=GetSystemMetrics(SM\_CYSCREEN);

//Регистрация оконного класса

wndClass.cbSize =sizeof(wndClass);

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

wndClass.lpfnWndProc =WndProc;

wndClass.cbClsExtra =0;

wndClass.cbWndExtra =0;

wndClass.hInstance =hInst;

wndClass.hIcon =LoadIcon(NULL,IDI\_WINLOGO);

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

wndClass.hbrBackground=GetStockBrush(BLACK\_BRUSH);

wndClass.lpszMenuName =NULL;

wndClass.lpszClassName=szClassName;

wndClass.hIconSm =LoadIcon(NULL,IDI\_WINLOGO);

RegisterClassEx(&wndClass);

//Создание окна на основе класса

hWnd=CreateWindowEx(

WS\_EX\_LEFT,//Дополнит. стиль окна

szClassName, //Класс окна

szAppName, //Текст заголовка

WS\_POPUP, //Стиль окна

0,0, //Координаты X и Y

nScreenX,nScreenY,//Ширина и высота

NULL, //Дескриптор родит. окна

NULL, //Дескриптор меню

hInst, //Описатель экземпляра

NULL); //Дополнит. данные

ShowWindow(hWnd,nCmdShow);

UpdateWindow(hWnd);

//Главный цикл программы

while (TRUE)

{

if (PeekMessage(&msg,NULL,0,0,PM\_REMOVE))

{

if (msg.message==WM\_QUIT)

break;

TranslateMessage(&msg);

DispatchMessage(&msg);

}

else

BMP\_OnIdle(hWnd);

}

return (msg.wParam);

}

/////////////////////////////////////////////////

// Оконная процедура

////////////////////////////////////////////////

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

{

switch(msg)

{

HANDLE\_MSG(hWnd,WM\_CREATE,BMP\_OnCreate);

HANDLE\_MSG(hWnd,WM\_DESTROY,BMP\_OnDestroy);

HANDLE\_MSG(hWnd,WM\_TIMER,BMP\_OnTimer);

HANDLE\_MSG(hWnd,WM\_KEYDOWN,BMP\_OnKey);

default:

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

}

}

/\* Обработчики сообщений \*/

BOOL BMP\_OnCreate(HWND hwnd, LPCREATESTRUCT lpCreateStruct)

{

HDC hTempDC;

//Получаем контекст Рабочего стола

hTempDC=GetDC(HWND\_DESKTOP);

//Создаём контекст в памяти и растр, совместимые с экраном

hMemDC=CreateCompatibleDC(hTempDC);

hMemBitmap=CreateCompatibleBitmap(hTempDC,nScreenX,nScreenY);

//Заливаем содержимое контекста черным цветом

SelectBitmap(hMemDC,hMemBitmap);

SelectBrush(hMemDC,GetStockBrush(BLACK\_BRUSH));

PatBlt(hMemDC,0,0,nScreenX,nScreenY,PATCOPY);

//Копируем картинку в контекст памяти

if (!LoadBMP(hMemDC,szFileName))

{

MessageBox(hwnd,"Не найден файл","Ошибка",MB\_OK|MB\_ICONSTOP);

return (FALSE);

}

//Освобождаем временный контекст

ReleaseDC(HWND\_DESKTOP,hTempDC);

//Создаём шрифт и переменные для цветов шрифта

hInfoFont=CreateFont(22,0,0,0,FW\_NORMAL,FALSE,FALSE,FALSE,

ANSI\_CHARSET,OUT\_DEFAULT\_PRECIS,CLIP\_DEFAULT\_PRECIS,

DEFAULT\_QUALITY,VARIABLE\_PITCH,"Courier New");

crFontColor1=RGB(255, 0,0);

crFontColor2=RGB(0, 255,0);

//Выбираем цвет шрифта

SetTextColor(hMemDC,crFontColor1);

//Рисуем текст в контексте памяти

ShowText();

//Устанавливаем таймер

if (!SetTimer(hwnd,TIMER\_ID,TIMER\_RATE,NULL))

return (FALSE);

return (TRUE);

}

//-----------------------------------------------------------------

void BMP\_OnDestroy(HWND hwnd)

{

//Убираем после себя

KillTimer(hwnd,TIMER\_ID);

DeleteDC(hMemDC);

DeleteBitmap(hMemBitmap);

DeleteFont(hInfoFont);

PostQuitMessage(0);

}

//-----------------------------------------------------------------

void BMP\_OnTimer(HWND hwnd, UINT id)

{

//Рисуем текст в контексте памяти

ShowText();

}

//-----------------------------------------------------------------

void BMP\_OnIdle(HWND hwnd)

{

HDC hWindowDC;

//Получаем контекст окна, копируем в него содержимое

//контекста памяти, освобождаем контекст окна

hWindowDC=GetDC(hwnd);

BitBlt(hWindowDC,0,0,nScreenX,nScreenY,hMemDC,0,0,SRCCOPY);

ReleaseDC(hwnd,hWindowDC);

}

//-----------------------------------------------------------------

void BMP\_OnKey(HWND hwnd, UINT vk, BOOL fDown, int cRepeat, UINT flags)

{

//При нажатии пробела - прекратить программу

if (vk==VK\_SPACE)

DestroyWindow(hwnd);

}

//-------------------------------------------------------------------

/\* Копирует содержимое файла BMP в контекст устройства \*/

BOOL LoadBMP(HDC hdc, char\* szFileName)

{

BYTE\* pBmp;

DWORD dwBmpSize;

DWORD dwFileLength;

DWORD dwBytesRead;

BITMAPFILEHEADER BmpHeader;

BITMAPINFO\* pBmpInfo;

BYTE\* pPixels;

//Пытаемся открыть файл

HANDLE hFile=CreateFile(szFileName, GENERIC\_READ, FILE\_SHARE\_READ, NULL, OPEN\_EXISTING, 0, NULL);

if (hFile==INVALID\_HANDLE\_VALUE)

return (FALSE);

//Определяем размер данных, которые необходимо загрузить

dwFileLength=GetFileSize (hFile, NULL) ;

dwBmpSize=dwFileLength-sizeof(BITMAPFILEHEADER);

//Выделяем память и считываем данные

pBmp= (BYTE\*) malloc(dwBmpSize);

ReadFile(hFile,&BmpHeader,sizeof(BmpHeader),&dwBytesRead,NULL);

ReadFile(hFile, (LPVOID)pBmp, dwBmpSize, &dwBytesRead,NULL);

CloseHandle(hFile);

//Инициализируем указатели на информацию о картинке

//и на графические данные

pBmpInfo=(BITMAPINFO\*)pBmp;

pPixels=pBmp+BmpHeader.bfOffBits-sizeof(BITMAPFILEHEADER);

//Получаем ширину и высоту картинки

nBmpHeight=pBmpInfo->bmiHeader.biHeight;

nBmpWidth=pBmpInfo->bmiHeader.biWidth;

//Вычистляем ширину и высоту картинки для вывода на экран

nVBmpHeight=nScreenY-nTopDown\*2;

nVBmpWidth=(UINT)((double)nBmpWidth\*(double)nVBmpHeight/(double)nBmpHeight);

if (nVBmpWidth>nScreenX)

nVBmpWidth=nScreenX;

//Устанавливаем режим масштабирования

SetStretchBltMode(hdc,HALFTONE);

//Копируем картинку в контекст памяти

StretchDIBits(hdc,(nScreenX-nVBmpWidth)/2,nTopDown, nVBmpWidth,nVBmpHeight,0,0,nBmpWidth,nBmpHeight,pPixels,pBmpInfo,0,SRCCOPY);

//Освобождаем память

free(pBmp);

return (TRUE);

}

/\* Меняет текущий цвет шрифта и выводит текст в контекст памяти\*/

void ShowText()

{

HFONT hOldFont;

RECT rTextRect;

char szInfoText[256];

//Определяем текущий цвет шрифта

if (GetTextColor(hMemDC)==crFontColor1)

SetTextColor(hMemDC,crFontColor2);

else

SetTextColor(hMemDC,crFontColor1);

//Производим настройки...

SetBkMode(hMemDC,TRANSPARENT);

hOldFont=SelectFont(hMemDC,hInfoFont);

//...и выводим текст.

SetRect(&rTextRect,0,0,nScreenX,nTopDown);

DrawText(hMemDC,szFileName,strlen(szFileName),&rTextRect,DT\_SINGLELINE|DT\_CENTER|DT\_VCENTER);

wsprintf(szInfoText, "Real width-%d, Real height-%d\n Virtual width-%d, Virtual height-%d",nBmpWidth,nBmpHeight,nVBmpWidth,nVBmpHeight);

SetRect(&rTextRect,0,nScreenY-nTopDown,nScreenX,nScreenY);

DrawText(hMemDC,szInfoText,strlen(szInfoText),&rTextRect,DT\_CENTER);

SelectFont(hMemDC,hOldFont);

}