**3 ЛАБОРАТОРНАЯ РАБОТА №3**

**ИНТЕРФЕЙС GDI. БИБЛИОТЕКА GDI+**

**Цель работы**

Изучить механизмы работы с интерфейсом графических устройств и библиотекой GDI+. Получить практические навыки по использования библиотеки GDI+.

**3.1 Листинг программы полученной в пункте 3.3**

#include <stdio.h>

#include <windows.h>

#include <gdiplus.h>

using namespace Gdiplus;

HDC memory\_dc;

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

HDC get\_memory\_dc(HWND hwnd){

HDC window\_dc = GetDC(hwnd);

HDC mem\_dc = CreateCompatibleDC(window\_dc);

if (mem\_dc == NULL){

return 0;

}

HBITMAP mem\_bitmap = CreateCompatibleBitmap(window\_dc, 500, 400);

SelectObject(mem\_dc, mem\_bitmap);

SelectObject(mem\_dc, GetStockObject(WHITE\_BRUSH));

PatBlt(mem\_dc, 0, 0, 500, 400, PATCOPY);

DeleteObject(mem\_bitmap);

ReleaseDC(hwnd, window\_dc);

return mem\_dc;

}

void draw\_all(HDC hdc, RECT rect){

WCHAR text[] = L"GDI+ example";

Graphics g(hdc);

g.SetPageUnit(UnitPixel);

RectF rectbounds(0, 0, float(rect.right), float(rect.bottom));

Image bg(L"TTL.gif");

g.DrawImage(&bg, rectbounds);

LinearGradientBrush brush(rectbounds, Color(255, 255, 0, 0),

Color(255, 0, 0, 255), LinearGradientModeHorizontal);

StringFormat format;

format.SetFormatFlags(StringFormatFlagsDirectionVertical);

format.SetAlignment(StringAlignmentFar);

format.SetLineAlignment(StringAlignmentCenter);

Font font(L"Arial", 48, FontStyleBold);

g.DrawString(text, 12, &font, rectbounds, &format, &brush);

}

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow){

   TCHAR class\_name[] = L"main window";

   TCHAR app\_title[] = L"simple\_window";

   WNDCLASS wc;

   HWND hWnd;

   MSG msg;

   memset(&wc, 0, sizeof(wc));

   wc.lpszClassName = class\_name;

   wc.lpfnWndProc = &WndProc;

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

   wc.hInstance = hInstance;

   wc.hIcon = LoadIcon(NULL, IDI\_APPLICATION);

   wc.hbrBackground = (HBRUSH)GetStockObject(WHITE\_BRUSH);

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

   wc.lpszMenuName = NULL;

   RegisterClass(&wc);

   hWnd = CreateWindow(class\_name,

   app\_title,

   WS\_OVERLAPPEDWINDOW,

   180,

   20,

   500,

   400,

   NULL,

   NULL,

   hInstance,

   NULL);

   if(!hWnd){

   MessageBox(NULL, L"cannot create window", app\_title, MB\_OK|MB\_ICONSTOP);

   return FALSE;

   }

   ShowWindow(hWnd, SW\_SHOWNORMAL);

   UpdateWindow(hWnd);

   GdiplusStartupInput GdiplusStartupInput;

   ULONG\_PTR gdiplusToken;

   GdiplusStartup(&gdiplusToken, &GdiplusStartupInput, NULL);

   memory\_dc = get\_memory\_dc(hWnd);

   RECT rect;

   GetClientRect(hWnd, &rect);

   draw\_all(memory\_dc, rect);

   while(GetMessage(&msg, hWnd, 0, 0)){

   TranslateMessage(&msg);

   DispatchMessage(&msg);

   }

   DeleteDC(memory\_dc);

   GdiplusShutdown(gdiplusToken);

   return msg.wParam;

}

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

TCHAR str[] = L"Text";

HDC hdc;

PAINTSTRUCT PaintStruct;

RECT rect;

switch(msg){

case WM\_PAINT:{

HDC hDC;

PAINTSTRUCT ps;

hDC = BeginPaint(hWnd, &ps);

BitBlt(hDC, ps.rcPaint.left, ps.rcPaint.top,

ps.rcPaint.right, ps.rcPaint.bottom,

memory\_dc, ps.rcPaint.left, ps.rcPaint.top, SRCCOPY);

EndPaint(hWnd, &ps);

break;

}

case WM\_DESTROY:

    PostQuitMessage(0);

break;

default:

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

}

return 0;

}

**3.2 Листинг программы заданной на самостоятельную работу**

#include <stdio.h>

#include <windows.h>

#include <windowsx.h>

#include <gdiplus.h>

using namespace Gdiplus;

HDC memory\_dc;

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

HDC get\_memory\_dc(HWND hwnd){

HDC window\_dc = GetDC(hwnd);

HDC mem\_dc = CreateCompatibleDC(window\_dc);

if (mem\_dc == NULL){

return 0;

}

HBITMAP mem\_bitmap = CreateCompatibleBitmap(window\_dc, 500, 400);

SelectObject(mem\_dc, mem\_bitmap);

SelectObject(mem\_dc, GetStockObject(WHITE\_BRUSH));

PatBlt(mem\_dc, 0, 0, 500, 400, PATCOPY);

DeleteObject(mem\_bitmap);

ReleaseDC(hwnd, window\_dc);

return mem\_dc;

}

void draw\_all(HDC hdc, RECT rect){

WCHAR text[] = L"GDI+ example";

Graphics g(hdc);

g.SetPageUnit(UnitPixel);

RectF rectbounds(0, 0, float(rect.right), float(rect.bottom));

Image bg(L"03.gif");

g.DrawImage(&bg, rectbounds);

LinearGradientBrush brush(rectbounds, Color(255, 255, 0, 0),

Color(255, 0, 0, 255), LinearGradientModeHorizontal);

StringFormat format;

format.SetFormatFlags(StringFormatFlagsDirectionVertical);

format.SetAlignment(StringAlignmentFar);

format.SetLineAlignment(StringAlignmentCenter);

Font font(L"Arial", 48, FontStyleBold);

g.DrawString(text, 12, &font, rectbounds, &format, &brush);

}

void draw\_rectangle(const RECT \*rect, HDC hDC){

Graphics g(hDC);

g.SetPageUnit(UnitPixel);

Pen pen(Color::Black, 2.0);

g.DrawRectangle(&pen, rect->left, rect->top, rect->right - rect->left, rect->bottom - rect->top);

}

void draw\_line(const RECT \*rect,HDC hDC){

Graphics g(hDC);

g.SetPageUnit(UnitPixel);

Image image(L"Bkg.gif");

TextureBrush brush(&image, WrapModeTile, Rect(90, 90, 90, 90));

Pen texturedPen(&brush, 35.0);

g.DrawLine(&texturedPen, rect->left, rect->top, rect->right, rect->bottom);

}

void draw\_ellipse(const RECT \*rect, HDC hDC){

Graphics g(hDC);

g.SetPageUnit(UnitPixel);

SolidBrush brush(Color::DarkOrchid);

g.FillEllipse(&brush, rect->left, rect->top, rect->right - rect->left, rect->bottom - rect->top);

}

void draw\_image(Image \*image, HDC hDC, const RECT \*rect){

Graphics g(hDC);

g.SetPageUnit(UnitPixel);

RectF rectbounds(0, 0, float(rect->right), float(rect->bottom));

g.DrawImage(image, rectbounds);

}

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

TCHAR str[] = L"Text";

HDC hdc;

PAINTSTRUCT PaintStruct;

RECT rect;

switch(msg){

case WM\_PAINT:{

HDC hDC;

PAINTSTRUCT ps;

hDC = BeginPaint(hWnd, &ps);

BitBlt(hDC, ps.rcPaint.left, ps.rcPaint.top,

ps.rcPaint.right, ps.rcPaint.bottom,

memory\_dc, ps.rcPaint.left, ps.rcPaint.top, SRCCOPY);

EndPaint(hWnd, &ps);

break;

}

case WM\_LBUTTONDOWN:{

if(wParam & MK\_SHIFT){

rect.left = LOWORD(lParam);

rect.top = HIWORD(lParam);

rect.right = LOWORD(lParam) + 50;

rect.bottom = HIWORD(lParam) + 90;

draw\_rectangle(&rect, memory\_dc);

InvalidateRect(hWnd, &rect, TRUE);

}

if(wParam & MK\_CONTROL){

rect.left = LOWORD(lParam);

rect.top = HIWORD(lParam);

rect.right = LOWORD(lParam) + 50;

rect.bottom = HIWORD(lParam) + 70;

draw\_line(&rect, memory\_dc);

InvalidateRect(hWnd, &rect, TRUE);

}

break;

}

case WM\_RBUTTONDOWN:

if(wParam & MK\_SHIFT){

rect.left = LOWORD(lParam);

rect.top = HIWORD(lParam);

rect.right = LOWORD(lParam) + 50;

rect.bottom = HIWORD(lParam) + 90;

draw\_ellipse(&rect, memory\_dc);

InvalidateRect(hWnd, &rect, TRUE);

}

if(wParam & MK\_CONTROL){

static Image bg(L"03.gif");

GetClientRect(hWnd, &rect);

bg.RotateFlip(Rotate90FlipNone);

draw\_image(&bg, memory\_dc, &rect);

InvalidateRect(hWnd, &rect, TRUE);

UpdateWindow(hWnd);

}

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

default:

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

}

return 0;

}

**Выводы**

В ходе лабораторной работы было написано приложение которое с помощью библиотеки может рисовать геометрические фигуры, такие как прямоугольник, текстурированная прямая, эллипс, а также выводить картинку в окно и поворачивать ее.