ЛАБОРАТОРНАЯ РАБОТА. РИСОВАНИЕ И ГРАФИКА

**Цель**. Рассмотреть средства графического вывода в MFC C++.

**Краткие сведения**.

Рисование «привязывается» к событию OnDraw сцены View. Используем графическую переменную (переменную графического контекста) CDC\* pDC. Она предоставляет методы для рисования.

Пояснения к программе рисования фигур

Рисование линий и геометрических фигур с заданным цветом выполняется программируемыми «перьями», а раскраска фигур осуществляется «кистями», которые реализуются в классах CBrush и СРеn.

Структура dwColor создается массив, в котором хранятся девять

RGB-значений цветов для используемых кистей и перьев:

static DWORD dwColor[9]=

{

RGB(0,0,0), //черный

RGB(255,0,0), //красный

RGB(0,255,0), //зеленый

RGBCO,0,255), //синий

RGB(255,255,0), //желтый

RGB(255,0,255), //пурпурный

RGB(0,255,255), //голубой

RGB(127,127,127), //серый

RGB(255,255,255) //белый

};

Кисти и перья создаются с помощью классов CBrush и СРеn, которые можно передавать функциям класса CDC - базового класса для контекста устройства отображения. Кисти могут иметь сплошную и штриховую заливку, а также заливку в виде растрового узора. Перья рисуют сплошной, штриховой или пунктирной линиями. Синтаксис программы,

отвечающей за создание кистей перьев, приведен ниже:

CBrush newbrush;

CBrush\* oldbrush;

CPen newpen;

CPen\* oldpen;

Поскольку для рисования различных графических примитивов применяются аналогичные алгоритмы, рассмотрим лишь два наиболее типичных фрагмента. В первом случае на экран выводится толстая черная диагональная линия.

// рисование толстой черной диагональной линии

newpen.CreatePen(PS\_SOLID,6,dwColor[0]);

oldpen-pDC->SelectObject(&newpen);

pDC->MoveTo(0,0);

pDC->LineTo(640,430);

pDC->TextOut(70,20,"<-diagonal 1ine",15);

// удаление пера

pDC->SelectObject(oldpen);

newpen.DeleteObjectO;

Перо создается функцией CreatePen(), которая задает рисование черных сплошных линий толщиной в шесть логических единиц. Сразу после этого функция SelectObject() обновляет перо, связывает его с контекстом устройства отображения (экраном монотора) и возвращает указатель на предыдущий объект пера. Функции LineTo() формируют диагональную линию, которая рисуется выбранными пером. Наконец, функция TextOut() выводит рядом с нарисованной фигурой надпись.

Работа с кистями организована аналогичным образом. В следующем коде создается кисть с заливкой в виде горизонтальных и вертикальных штрихов (HS\_CROSS) синего цвета:

// рисование черного закругленного прямоугольника

// и заливка его синим цветом

newbrush.CreateHatchBrush(HS\_CROSS,dwColor[3]);

oldbrush=pDC->SelectObject(&newbrush);

pDC->RoundRect(60,310,110,350,20,20);

pDC->TextOut (120,310,"<-rounded rectangle",19);

// удаление кисти

pDC->SelectObject(oldbrush);

newbrush.DeleteObject С); 66

Функция RoundRect () рисует прямоугольник с закругленными

краями и заданными координатами.

Полный текст программы:

// Graphic\_CView.cpp : implementation of the CGraphic\_CView class

//

#include "stdafx.h"

// SHARED\_HANDLERS can be defined in an ATL project implementing preview, thumbnail

// and search filter handlers and allows sharing of document code with that project.

#ifndef SHARED\_HANDLERS

#include "Graphic\_C.h"

#endif

#include "Graphic\_CDoc.h"

#include "Graphic\_CView.h"

#ifdef \_DEBUG

#define new DEBUG\_NEW

#endif

// CGraphic\_CView

IMPLEMENT\_DYNCREATE(CGraphic\_CView, CView)

BEGIN\_MESSAGE\_MAP(CGraphic\_CView, CView)

// Standard printing commands

ON\_COMMAND(ID\_FILE\_PRINT, &CView::OnFilePrint)

ON\_COMMAND(ID\_FILE\_PRINT\_DIRECT, &CView::OnFilePrint)

ON\_COMMAND(ID\_FILE\_PRINT\_PREVIEW, &CGraphic\_CView::OnFilePrintPreview)

ON\_WM\_CONTEXTMENU()

ON\_WM\_RBUTTONUP()

END\_MESSAGE\_MAP()

// CGraphic\_CView construction/destruction

CGraphic\_CView::CGraphic\_CView()

{

// TODO: add construction code here

}

CGraphic\_CView::~CGraphic\_CView()

{

}

BOOL CGraphic\_CView::PreCreateWindow(CREATESTRUCT& cs)

{

// TODO: Modify the Window class or styles here by modifying

// the CREATESTRUCT cs

return CView::PreCreateWindow(cs);

}

// CGraphic\_CView drawing

void CGraphic\_CView::OnDraw(CDC\* pDC)

{

// CGraphic\_CDoc\* pDC = GetDocument();

// CGdiDoc\* pDoc = GetDocument( );

ASSERT\_VALID(pDC);

static DWORD dwColor[9]={

RGB(0,0,0),

//черный

RGB(255,0,0), //красный

RGB(0,255,0), //зеленый

RGB(0,0,255), //синий

RGB(255,255,0), //желтый

RGB(255,0,255), //пурпурный

RGB(0,255,255), //голубой

RGB(127,127,127), //серый

RGB(255,255,255)}; //белый

int xcoord;

POINT polylpts[4],polygpts[5];

CBrush newbrush;

CBrush\* oldbrush;

CPen newpen;

CPen\* oldpen;

// рисование толстой диагональной линии

newpen.CreatePen(PS\_SOLID,6,dwColor[0]);

oldpen=pDC->SelectObject(&newpen);

pDC->MoveTo(0,0);

pDC->LineTo(640,430);

//pDC->TextOut(70,20,"<-diagonalline",15);

// удаление пера

pDC->SelectObject (oldpen);

newpen.DeleteObject();

// рисование синей дуги

newpen.CreatePen(PS\_DASH,1,dwColor[3]);

oldpen=pDC->SelectObject(&newpen);

pDC->Arc(100,100,200,200,150,175,175,150);

//pDC->TextOut(80,180,"smal1 arc->",11);

// удаление пера

pDC->SelectObject(oldpen);

newpen.DeleteObject( );

// рисование сегмента с толстым зеленым контуром

newpen.CreatePen(PS\_SOLID,8,dwColor[2]);

oldpen=pDC->SelectObject(&newpen);

pDC->Chord(550,20,630,80,555,25,625,70);

//pDC->TextOut(485,30,"chord->",7);

// удаление пера

pDC->SelectObject(oldpen);

newpen.DeleteObject ( );

// рисование эллипса и заливка его красным цветом

CreatePen(PS\_SOLID,1,dwColor[1]);

oldpen=pDC->SelectObject(&newpen);

newbrush.CreateSolidBrush(dwColor[1]);

oldbrush=pDC->SelectObject(&newbrush);

pDC->Ellipse(180,180,285,260);

//pDC->TextOut(210,215, "ellipse", 7);

// удаление кисти

pDC->SelectObject(oldbrush);

newbrush.DeleteObject();

// удаление пера

pDC->SelectObject(oldpen);

newpen. DeleteObject ( );

// рисование круга и заливка его синим цветом

newpen.CreatePen(PS\_SOLID,1,dwColor[3]);

oldpen=pDC->SelectObject(&newpen);

newbrush.CreateSolidBrush (dwColor[3]);

oldbrush=pDC->SelectObject(&newbrush);

pDC->Ellipse(380,180,570,370);

//pDC->TextOut(450,265,"circle",6);

// удаление кисти

pDC->SelectObject(oldbrush);

newbrush.DeleteObject() ;

// удаление пера

pDC->SelectObject(oldpen) ;

newpen.DeleteObject();

// рисование сектора и заливка его зеленым цветом

newpen.CreatePen(PS\_SOLID,1,dwColor[0]);

oldpen=pDC->SelectObject(&newpen);

newbrush.CreateSolidBrush(dwColor[2]);

oldbrush=pDC->SelectObject(&newbrush);

pDC->Pie(300,50,400,150,300,50,300,100);

pDC->TextOut(350,80,TEXT("<-pie wedge"),11);

// удаление кисти

pDC->SelectObject(oldbrush) ;

newbrush.DeleteObject();

//удаление пера

pDC->SelectObject(oldpen) ;

newpen.DeleteObject();

// рисование прямоугольника и заливка его серым цветом

newbrush.CreateSolidBrush(dwColor[7]);

oldbrush=pDC->SelectObject(&newbrush);

pDC->Rectangle(50,300,150,400);

pDC->TextOut(160,350,TEXT("<-rectangle"),11);

// удаление кисти

pDC->SelectObject(oldbrush);

newbrush.DeleteObject( );

// рисование закругленного прямоугольника

// и заливка его синим цветом

newbrush.CreateHatchBrush(HS\_CROSS,dwColor[3]);

oldbrush=pDC->SelectObject(&newbrush);

pDC->RoundRect(60,310,110,350,20,20);

pDC->TextOut (120,310,TEXT("<-rounded rectangle") ,19);

// удаление кисти

pDC->SelectObject(oldbrush);

newbrush.DeleteObject( );

// рисование нескольких точеск

for(xcoord=400;xcoord<450;xcoord+=3)

pDC->SetPixel(xcoord,150,0L);

//pDC->TextOut(455,145,"<-pixels",8);

// рисование толстой ломаной линии пурпурного цвета

newpen.CreatePen(PS\_SOLID,3,dwColor[5]); oldpen=pDC->SelectObject(&newpen);

polylpts[0].x=10;

polylpts[0].y=30;

polylpts[1].x=10;

polylpts[1].y=100;

polylpts[2].x=50;

polylpts[2].y=100;

polylpts[3].x=10;

polylpts[3].y=30;

pDC->Polyline(polylpts,4);

pDC->TextOut(10,110,TEXT("polyline"),8);

// удаление пера

pDC->SelectObject(oldpen);

newpen.DeleteObject( );

// рисование многоугольника с голубым контуром

// и заливка его диагональной желтой штриховкой

newpen.CreatePen(PS\_SOLID,4,dwColor[6]);

oldpen=pDC->SelectObject(&newpen);

newbrush.CreateHatchBrush(HS\_FDIAGONAL,dwColor[4]);

oldbrush=pDC->SelectObject(&newbrush);

polygpts[0].x=40;

polygpts[0].y=200;

polygpts[1].x=100;

polygpts[1].y=270;

polygpts[2].x=80;

polygpts[2].y=290;

polygpts[3].x=20;

polygpts[3].y=220;

polygpts[4].x=40;

polygpts[4].y=200;

pDC->Polygon(polygpts,5);

pDC->TextOut(70,210,TEXT("<-polygon"),9);

// удаление кисти

pDC->SelectObject(oldbrush);

newbrush.DeleteObject ();

// удаление пера

pDC-> SelectObject(oldpen);

newpen.DeleteObject( );

}

// CGraphic\_CView printing

void CGraphic\_CView::OnFilePrintPreview()

{

#ifndef SHARED\_HANDLERS

AFXPrintPreview(this);

#endif

}

BOOL CGraphic\_CView::OnPreparePrinting(CPrintInfo\* pInfo)

{

// default preparation

return DoPreparePrinting(pInfo);

}

void CGraphic\_CView::OnBeginPrinting(CDC\* /\*pDC\*/, CPrintInfo\* /\*pInfo\*/)

{

// TODO: add extra initialization before printing

}

void CGraphic\_CView::OnEndPrinting(CDC\* /\*pDC\*/, CPrintInfo\* /\*pInfo\*/)

{

// TODO: add cleanup after printing

}

void CGraphic\_CView::OnRButtonUp(UINT /\* nFlags \*/, CPoint point)

{

ClientToScreen(&point);

OnContextMenu(this, point);

}

void CGraphic\_CView::OnContextMenu(CWnd\* /\* pWnd \*/, CPoint point)

{

#ifndef SHARED\_HANDLERS

theApp.GetContextMenuManager()->ShowPopupMenu(IDR\_POPUP\_EDIT, point.x, point.y, this, TRUE);

#endif

}

// CGraphic\_CView diagnostics

#ifdef \_DEBUG

void CGraphic\_CView::AssertValid() const

{

CView::AssertValid();

}

void CGraphic\_CView::Dump(CDumpContext& dc) const

{

CView::Dump(dc);

}

CGraphic\_CDoc\* CGraphic\_CView::GetDocument() const // non-debug version is inline

{

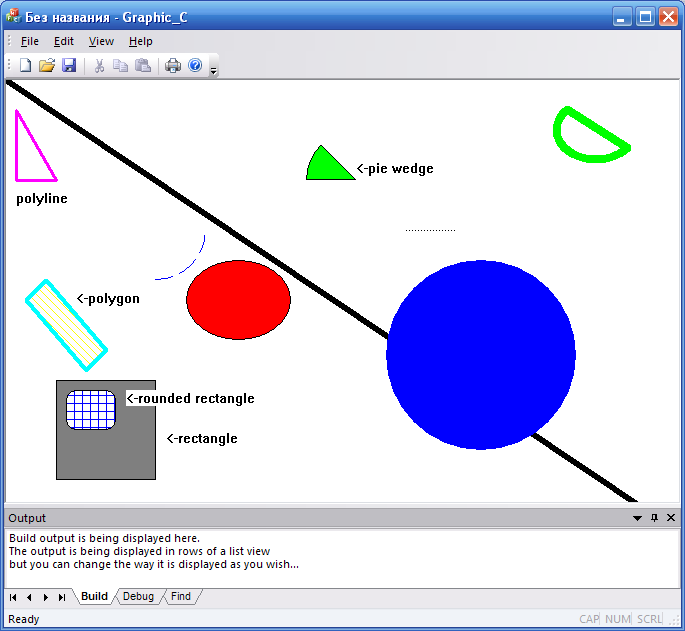
ASSERT(m\_pDocument->IsKindOf(RUNTIME\_CLASS(CGraphic\_CDoc)));

return (CGraphic\_CDoc\*)m\_pDocument;

}

#endif //\_DEBUG

// CGraphic\_CView message handlers



**ЗАДАНИЕ**.

1. Построить график синусоиды
2. Построить график спирали
3. Построить график произвольной функции, задаваемой массивом точек.