MFC画图板程序

骆克云 2015.12.18

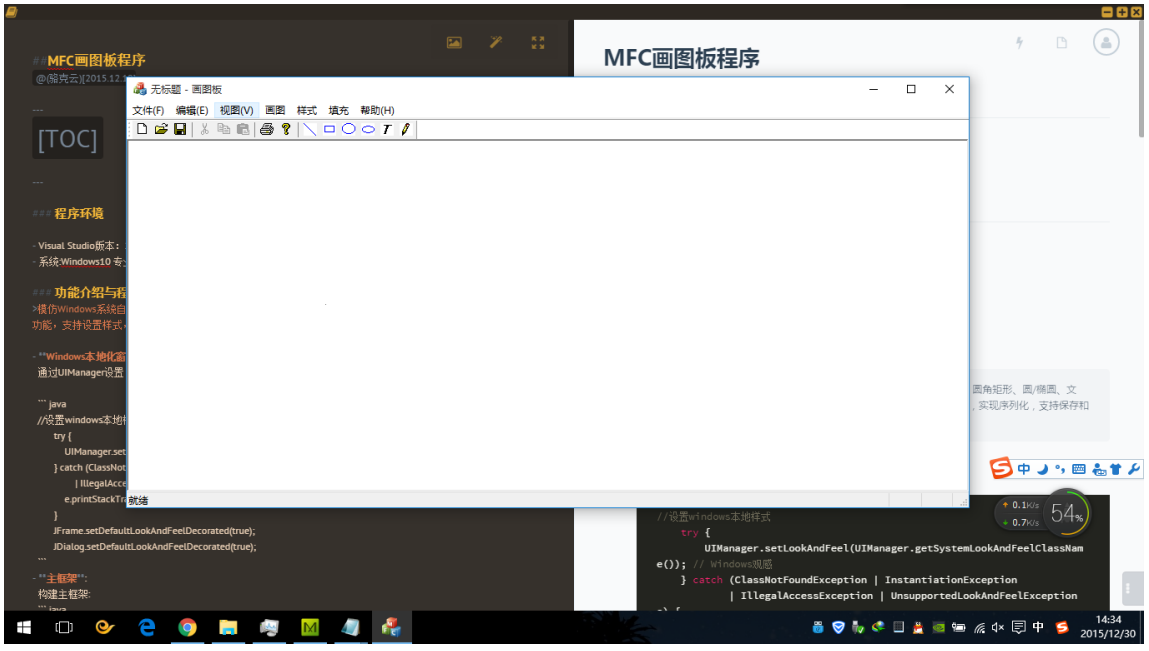
MFC画图板程序
程序环境
运行截图
功能介绍与程序思想

程序环境

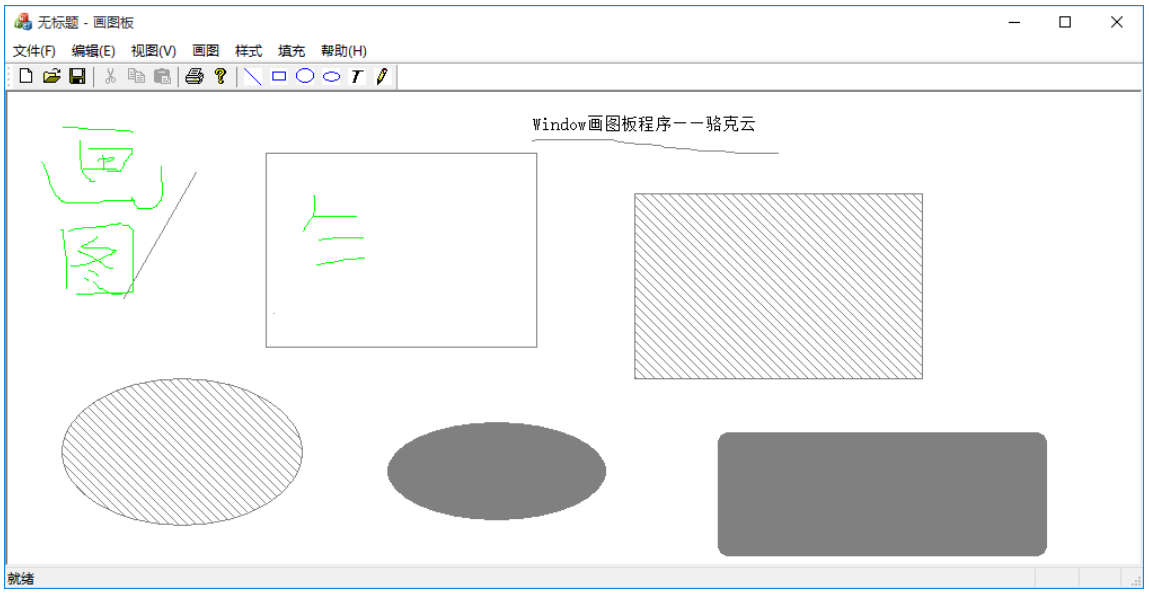
- Visual Studio版本：2015企业版
- 系统:Windows10 专业版64位

运行截图

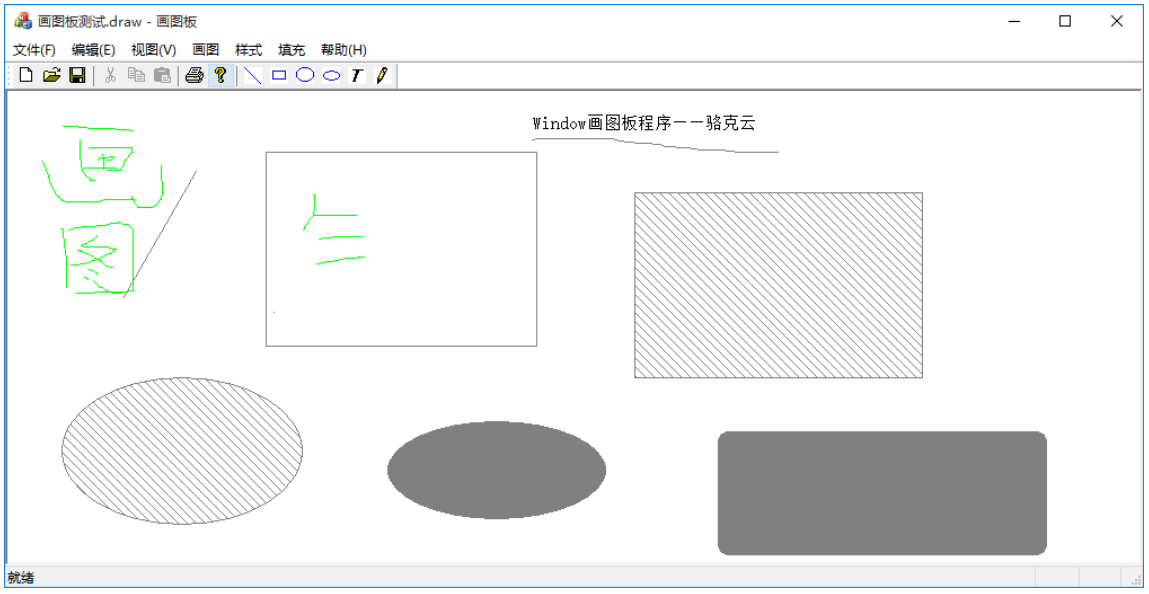
- 总体框架



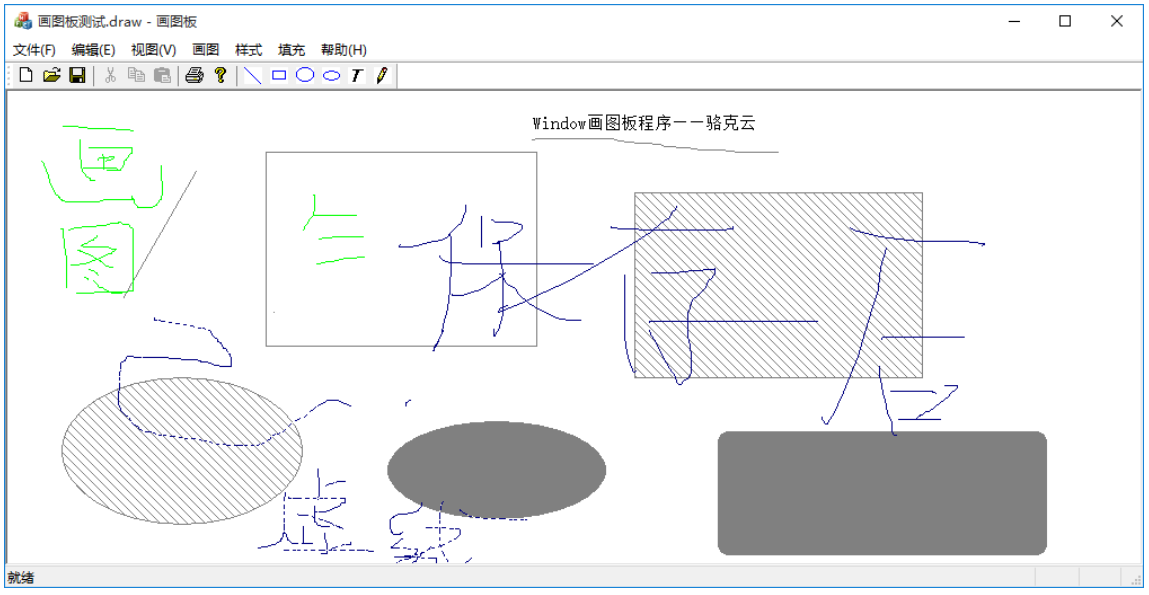
- 绘制图形文字样式



- 保存后打开文件



- 绘制虚线



功能介绍与程序思想

> 模仿Windows系统自带的画图板程序。基于MVC模型，实现直线、直角矩形、圆角矩形、圆/椭圆、文字、铅笔等功能，支持设置样式，包括线宽、颜色、填充方式、边框、字体等，实现序列化，支持保存和存档后加载。

- **序列化类**:
  类定义：必须继承自`CObject`

```
 class Shapes :public CObject
{
DECLARE_SERIAL(Shapes)
public:
int        m_DrawingType;//标志画的是矩形、直线、椭圆等类型

COLORREF   m_penColor;//颜色序列化
int        m_penStyle;
int        m_penWide;
int        m_brushStyle;
COLORREF   m_brushColor;         //画刷颜色序列化

int        m_pictureFourStyle;/*针对封闭图形的四种方式
                              即只有边框、
                              有边框的6种线条填充(又包含6种)、
                              边框和填充、
                              无边框
                              的标志*/
CPoint     m_startPoint;
CPoint     m_endPoint;
CString    m_Text;//记录写入的字符
LOGFONT    m_logFont;
COLORREF   m_fontColor;
Shapes();
virtual void Serialize(CArchive& ar);
virtual ~Shapes();
};
```

序列化实现：重写Serialize类

```
void Shapes::Serialize(CArchive& ar)
{
    CObject::Serialize(ar);
    if (ar.IsStoring())
    {
        ar << m_DrawingType << m_startPoint << m_endPoint;//存储图形的位
置,及种类
        if (m_DrawingType == 5)//字体存储
        {
            ar << m_logFont.lfHeight << m_logFont.lfWidth << m_logFont.l
fEscapement
                << m_logFont.lfOrientation << m_logFont.lfWeight << m_lo
gFont.lfItalic
                << m_logFont.lfUnderline << m_logFont.lfStrikeOut << m_l
ogFont.lfCharSet
                << m_logFont.lfOutPrecision << m_logFont.lfClipPrecision
<< m_logFont.lfQuality
                << m_logFont.lfPitchAndFamily;
            int i = 0;
            for (i = 0; i < LF_FACESIZE; i++)//LF_FACESIZE=32
            {
                ar << m_logFont.lfFaceName[i];//char数组要把每个元素一一取出
分别存储
            }
            ar << m_Text;
            ar << m_fontColor;
            return;
        }
        ar << m_penStyle << m_penWide << m_penColor;//ar<<m_Pen;只能存储基
本的数据类型
        if (m_DrawingType == 1 || m_DrawingType == 6)
        {
            ar << m_startPoint << m_endPoint << (DWORD)m_penColor;
            return;
        }
        else
        {
            ar << m_pictureFourStyle << m_brushStyle << m_brushColor;//存
储图形样式及画刷
        }
    }
    else
    {
        ar >> m_DrawingType >> m_startPoint >> m_endPoint;//获取图形的位
置,及种类
        if (m_DrawingType == 5)//字体存储
        {
            ar >> m_logFont.lfHeight >> m_logFont.lfWidth >> m_logFont.l
fEscapement
                >> m_logFont.lfOrientation >> m_logFont.lfWeight >> m_lo
```

```
gFont.lfItalic
                >> m_logFont.lfUnderline >> m_logFont.lfStrikeOut >> m_l
ogFont.lfCharSet
                >> m_logFont.lfOutPrecision >> m_logFont.lfClipPrecision
>> m_logFont.lfQuality
                >> m_logFont.lfPitchAndFamily;
            int i = 0;
            for (i = 0; i < LF_FACESIZE; i++)//sizeof(m_logFont.lfFaceNa
me
            {
                ar >> m_logFont.lfFaceName[i];//char数组要把每个元素一一取出
分别存储
            }
            ar >> m_Text;
            ar >> m_fontColor;
            return;
        }
        ar >> m_penStyle >> m_penWide >> m_penColor;
        if (m_DrawingType == 1 || m_DrawingType == 6)
        {
            ar >> m_startPoint >> m_endPoint >> (DWORD)m_penColor;
            return;
        }
        else
        {
            ar >> m_pictureFourStyle >> m_brushStyle >> m_brushColor;//获
取图形样式及画刷
        }
    }
}
```

- **Draw函数**:
  分类别绘制:

```
 // CMFCDrawView 绘制

void CMFCDrawView::OnDraw(CDC*  pDC)
{
CMFCDrawDoc* pDoc = GetDocument();
ASSERT_VALID(pDoc);
if (!pDoc)
    return;

POSITION pos = pDoc->shapeArray.GetHeadPosition();
while (pos != NULL)
{
    Shapes *shp = pDoc->shapeArray.GetNext(pos);
    if (shp->m_DrawingType == 1)
    {
        CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penColor);
        pDC->SelectObject(&pen);
        pDC->MoveTo(shp->m_startPoint);
        pDC->LineTo(shp->m_endPoint);
    }
    else if (shp->m_DrawingType == 2)
    {
        if (shp->m_pictureFourStyle == 1)//有边框，无填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            pDC->SelectObject(&pen);
            pDC->SelectStockObject(NULL_BRUSH);
            pDC->Rectangle(shp->m_startPoint.x, shp->m_startPoin
t.y, shp->m_endPoint.x, shp->m_endPoint.y);
        }
        else if (shp->m_pictureFourStyle == 2)//有边框，白色填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            CBrush brush(shp->m_brushColor);
            pDC->SelectObject(&pen);
            pDC->SelectObject(&brush);
            pDC->Rectangle(shp->m_startPoint.x, shp->m_startPoin
t.y, shp->m_endPoint.x, shp->m_endPoint.y);
        }
        else if (shp->m_pictureFourStyle == 3)//无边框，颜色填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            CBrush brush(shp->m_brushColor);
            pDC->SelectObject(&pen);
            pDC->SelectObject(&brush);
            pDC->Rectangle(shp->m_startPoint.x, shp->m_startPoin
t.y, shp->m_endPoint.x, shp->m_endPoint.y);
```

```
        }
        else if (shp->m_pictureFourStyle == 4)//有边框，线条填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            CBrush brush(shp->m_brushStyle, shp->m_brushColor);
            pDC->SelectObject(&pen);
            pDC->SelectObject(&brush);
            pDC->Rectangle(shp->m_startPoint.x, shp->m_startPoin
t.y, shp->m_endPoint.x, shp->m_endPoint.y);
        }

    }
    else if (shp->m_DrawingType == 3)
    {
        if (shp->m_pictureFourStyle == 1)//有边框，无填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            pDC->SelectObject(&pen);
            pDC->SelectStockObject(NULL_BRUSH);
            pDC->RoundRect(shp->m_startPoint.x, shp->m_startPoin
t.y, shp->m_endPoint.x, shp->m_endPoint.y, 18, 18);
        }
        else if (shp->m_pictureFourStyle == 2)//有边框，白填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            CBrush brush(shp->m_brushColor);
            pDC->SelectObject(&pen);
            pDC->SelectObject(&brush);
            pDC->RoundRect(shp->m_startPoint.x, shp->m_startPoin
t.y, shp->m_endPoint.x, shp->m_endPoint.y, 18, 18);
        }
        else if (shp->m_pictureFourStyle == 3)//无边框，颜色填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            CBrush brush(shp->m_brushColor);
            pDC->SelectObject(&pen);
            pDC->SelectObject(&brush);
            pDC->RoundRect(shp->m_startPoint.x, shp->m_startPoin
t.y, shp->m_endPoint.x, shp->m_endPoint.y, 18, 18);
        }
        else if (shp->m_pictureFourStyle == 4)//有边框，线条填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            CBrush brush(shp->m_brushStyle, shp->m_brushColor);
            pDC->SelectObject(&pen);
            pDC->SelectObject(&brush);
```

```
            pDC->RoundRect(shp->m_startPoint.x, shp->m_startPoin
t.y, shp->m_endPoint.x, shp->m_endPoint.y, 18, 18);
        }
    }
    else if (shp->m_DrawingType == 4)//画椭圆
    {
        if (shp->m_pictureFourStyle == 1)//有边框，无填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            pDC->SelectObject(&pen);
            pDC->SelectStockObject(NULL_BRUSH);
            pDC->Ellipse(shp->m_startPoint.x, shp->m_startPoint.y,
shp->m_endPoint.x, shp->m_endPoint.y);
        }
        else if (shp->m_pictureFourStyle == 2)//有边框，白填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            CBrush brush(shp->m_brushColor);
            pDC->SelectObject(&pen);
            pDC->SelectObject(&brush);
            pDC->Ellipse(shp->m_startPoint.x, shp->m_startPoint.y,
shp->m_endPoint.x, shp->m_endPoint.y);
        }
        else if (shp->m_pictureFourStyle == 3)//无边框，颜色填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            CBrush brush(shp->m_brushColor);
            pDC->SelectObject(&pen);
            pDC->SelectObject(&brush);
            pDC->Ellipse(shp->m_startPoint.x, shp->m_startPoint.y,
shp->m_endPoint.x, shp->m_endPoint.y);
        }
        else if (shp->m_pictureFourStyle == 4)//有边框，线条填充
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penCol
or);
            CBrush brush(shp->m_brushStyle, shp->m_brushColor);
            pDC->SelectObject(&pen);
            pDC->SelectObject(&brush);
            pDC->Ellipse(shp->m_startPoint.x, shp->m_startPoint.y,
shp->m_endPoint.x, shp->m_endPoint.y);
        }
    }
    else if (shp->m_DrawingType == 5)//画字符
    {
        pFont = new CFont();
        pFont->CreateFontIndirect(&shp->m_logFont);
        pDC->SetBkMode(TRANSPARENT);
```

```
            pDC->SetTextColor(shp->m_fontColor);
            pDC->SelectObject(pFont);
            pDC->TextOut(shp->m_startPoint.x, shp->m_startPoint.y, shp-
>m_Text);
        }

        else if (shp->m_DrawingType == 6)//画铅笔字
        {
            CPen pen(shp->m_penStyle, shp->m_penWide, shp->m_penColor);
            pDC->SelectObject(&pen);
            pDC->MoveTo(shp->m_startPoint);
            pDC->LineTo(shp->m_endPoint);
        }
    }
    }
```

- **设置字体**

```
void CMFCDrawView::OnFont()
{
CFontDialog fontDialog(&logFont);
int result = fontDialog.DoModal();

if (result == IDOK)
{
    pFont = new CFont();
    pFont->CreateFontIndirect(&logFont);
    fontColor = fontDialog.GetColor();
    MessageBox(L"字体设置成功");
    return;
}
MessageBox(L"取消字体设置");
}
```

- **设置线宽**

```
void CMFCDrawView::OnChangeLineWide(UINT uID)
{
    switch (uID)
    {
    case ID_LINE_MIN://细
        penWide = 1;
        break;
    case ID_LINE_MID://中
        penWide = 5;
        break;
    case ID_LINE_WIDE://宽
        penWide = 18;
        break;
    case ID_LINE_ADDWIDE://加宽
        penWide++;
        if (penWide > 30)//设置最宽为30个像素
        {
            penWide = 1;
        }
        break;
    }
}
```

- **鼠标左键响应**

```
void CMFCDrawView::OnLButtonDown(UINT nFlags, CPoint point)
{
    CMFCDrawDoc* pDoc = GetDocument();
    SetCapture();
    startPoint = oldEndPoint = newEndPoint = point;
    if (DrawingType == 5)//写字
    {
        InputDlg dlg;
        int result = dlg.DoModal();
        if (result == IDOK)
        {
            Shapes *sh = new Shapes();
            CString s = dlg.m_Edit;
            CDC*pDC = GetDC();
            //设置文字背景透明
            pDC->SetBkMode(TRANSPARENT);
            pDC->SelectObject(pFont);
            pDC->SetTextColor(fontColor);
            pDC->TextOut(point.x, point.y, s);
            if (!s.IsEmpty())
            {
                sh->m_DrawingType = DrawingType;
                sh->m_startPoint = point;
                sh->m_logFont = logFont;
                sh->m_fontColor = fontColor;
                sh->m_Text = s;
                pDoc->shapeArray.AddTail(sh);
                pDoc->SetModifiedFlag();
            }
        }
    }
    CView::OnLButtonDown(nFlags, point);
}


void CMFCDrawView::OnLButtonUp(UINT nFlags, CPoint point)
{
    ReleaseCapture();
    CDC *pDC = GetDC();
    CMFCDrawDoc *pDoc = GetDocument();
    if (clickAndMove)
    {
        Shapes *shap = new Shapes();
        CPen pen(penStyle, penWide, penColor);
        pDC->SelectObject(&pen);

        if (DrawingType == 1)//直线
        {
            pDC->MoveTo(startPoint);
            pDC->LineTo(newEndPoint);
```

```
        }
        else if (DrawingType == 2)//矩形
        {
            if (pictureFourStyle == 1)//有边框，无填充
            {
                pDC->SelectStockObject(NULL_BRUSH);
                pDC->Rectangle(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y);
            }
            else if (pictureFourStyle == 2)//有边框，白填充
            {
                brushColor = RGB(255, 255, 255);
                pDC->Rectangle(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y);
            }
            else if (pictureFourStyle == 3)//有边框，颜色填充
            {

                pDC->SelectObject(&pen);
                CBrush brush(brushColor);
                pDC->SelectObject(&brush);
                pDC->Rectangle(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y);
            }
            else if (pictureFourStyle == 4)//有边框，线条填充
            {
                brushColor = penColor;
                CBrush brush(brushStyle, brushColor);
                pDC->SelectObject(&brush);
                pDC->Rectangle(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y);
            }
        }
        else if (DrawingType == 3)//圆角矩形
        {
            if (pictureFourStyle == 1)//有边框，无填充
            {
                pDC->SelectStockObject(NULL_BRUSH);
                pDC->RoundRect(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y, 18, 18);
            }
            else if (pictureFourStyle == 2)//有边框，白填充
            {
                brushColor = RGB(255, 255, 255);
                shap->m_brushColor = RGB(255, 255, 255);
                pDC->RoundRect(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y, 18, 18);
            }
            else if (pictureFourStyle == 3)//有边框，颜色填充
            {
                brushColor = penColor;
```

```
                CBrush brush(brushColor);
                pDC->SelectObject(&brush);
                pDC->RoundRect(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y, 18, 18);
            }
            else if (pictureFourStyle == 4)//有边框，线条填充
            {
                brushColor = penColor;
                CBrush brush(brushStyle, brushColor);
                pDC->SelectObject(&brush);
                pDC->RoundRect(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y, 18, 18);
            }
        }
        else if (DrawingType == 4)//椭圆
        {
            if (pictureFourStyle == 1)//有边框，无填充
            {
                pDC->SelectStockObject(NULL_BRUSH);
                pDC->Ellipse(startPoint.x, startPoint.y, newEndPoint.x,
newEndPoint.y);
            }
            else if (pictureFourStyle == 2)//有边框，白色填充
            {
                brushColor = RGB(255, 255, 255);
                pDC->Ellipse(startPoint.x, startPoint.y, newEndPoint.x,
newEndPoint.y);
            }
            else if (pictureFourStyle == 3)//有边框，颜色填充
            {
                brushColor = penColor;
                CBrush brush(brushColor);
                pDC->SelectObject(&brush);

                pDC->Ellipse(startPoint.x, startPoint.y, newEndPoint.x,
newEndPoint.y);
            }
            else if (pictureFourStyle == 4)//有边框，线条填充
            {
                brushColor = penColor;
                CBrush brush(brushStyle, brushColor);
                pDC->SelectObject(&brush);
                pDC->Ellipse(startPoint.x, startPoint.y, newEndPoint.x,
newEndPoint.y);
            }
        }
        else if (DrawingType == 6)
        {
            CView::OnLButtonUp(nFlags, point);
            return;
        }
```

```
        shap->m_DrawingType = DrawingType;
        shap->m_startPoint = startPoint;
        shap->m_endPoint = newEndPoint;
        shap->m_penStyle = penStyle;
        shap->m_penWide = penWide;
        shap->m_penColor = penColor;
        //如果保存直线，只要保存上面信息已经足够
        shap->m_pictureFourStyle = pictureFourStyle;
        shap->m_brushStyle = brushStyle;
        shap->m_brushColor = brushColor;
        clickAndMove = FALSE;
        pDoc->shapeArray.AddTail(shap);
        pDoc->SetModifiedFlag();
    }
    CView::OnLButtonUp(nFlags, point);
}


void CMFCDrawView::OnMouseMove(UINT nFlags, CPoint point)
{
    newEndPoint = point;
    CDC*pDC = GetDC();
    if (nFlags&MK_LBUTTON)
    {
        clickAndMove = TRUE;
        if (DrawingType == 1)//直线
        {
            CPen pen(penStyle, penWide, penColor);
            pDC->SelectObject(&pen);

            pDC->SetROP2(R2_XORPEN);

            pDC->MoveTo(startPoint);
            pDC->LineTo(oldEndPoint);

            pDC->MoveTo(startPoint);
            pDC->LineTo(newEndPoint);
        }
        else if (DrawingType == 2)//直角矩形
        {
            if (pictureFourStyle == 1)//有边框，无填充
            {
                // MessageBox("直角矩形1");
                CPen pen(penStyle, penWide, penColor);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->SelectStockObject(NULL_BRUSH);

                pDC->Rectangle(startPoint.x, startPoint.y, oldEndPoin
t.x, oldEndPoint.y);
```

```
                pDC->Rectangle(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y);
            }
            else if (pictureFourStyle == 2)//有边框，白色填充
            {
                CPen pen(penStyle, penWide, penColor);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->Rectangle(startPoint.x, startPoint.y, oldEndPoin
t.x, oldEndPoint.y);
                pDC->Rectangle(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y);
            }
            else if (pictureFourStyle == 3)//有边框，颜色填充
            {
                CPen pen(penStyle, penWide, penColor);
                CBrush brush(brushColor);
                pDC->SelectObject(&brush);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->Rectangle(startPoint.x, startPoint.y, oldEndPoin
t.x, oldEndPoint.y);
                pDC->Rectangle(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y);
            }
            else if (pictureFourStyle == 4)//有边框，线条填充
            {
                brushColor = penColor;
                CPen pen(penStyle, penWide, penColor);
                CBrush brush(brushStyle, brushColor);
                pDC->SelectObject(&brush);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->Rectangle(startPoint.x, startPoint.y, oldEndPoin
t.x, oldEndPoint.y);
                pDC->Rectangle(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y);
            }
        }
        else if (DrawingType == 3)//圆角矩形
        {
            if (pictureFourStyle == 1)//有边框，无填充
            {
                CPen pen(penStyle, penWide, penColor);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->SelectStockObject(NULL_BRUSH);

                pDC->RoundRect(startPoint.x, startPoint.y, oldEndPoin
t.x, oldEndPoint.y, 18, 18);
                pDC->RoundRect(startPoint.x, startPoint.y, newEndPoin
```

```
t.x, newEndPoint.y, 18, 18);
                //MessageBox("圆角矩形1");
            }
            else if (pictureFourStyle == 2)//有边框，白填充
            {
                CPen pen(penStyle, penWide, penColor);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->RoundRect(startPoint.x, startPoint.y, oldEndPoin
t.x, oldEndPoint.y, 18, 18);
                pDC->RoundRect(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y, 18, 18);
            }
            else if (pictureFourStyle == 3)//有边框，颜色填充
            {
                CPen pen(penStyle, penWide, penColor);
                CBrush brush(brushColor);
                pDC->SelectObject(&brush);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->RoundRect(startPoint.x, startPoint.y, oldEndPoin
t.x, oldEndPoint.y, 18, 18);
                pDC->RoundRect(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y, 18, 18);
            }
            else if (pictureFourStyle == 4)//有边框，线条填充
            {
                brushColor = penColor;
                CPen pen(penStyle, penWide, penColor);
                CBrush brush(brushStyle, brushColor);
                pDC->SelectObject(&brush);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->RoundRect(startPoint.x, startPoint.y, oldEndPoin
t.x, oldEndPoint.y, 18, 18);
                pDC->RoundRect(startPoint.x, startPoint.y, newEndPoin
t.x, newEndPoint.y, 18, 18);
            }
        }
        else if (DrawingType == 4)//椭圆
        {
            if (pictureFourStyle == 1)//有边框，无填充
            {
                CPen pen(penStyle, penWide, penColor);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->SelectStockObject(NULL_BRUSH);

                pDC->Ellipse(startPoint.x, startPoint.y, oldEndPoint.x,
oldEndPoint.y);
                pDC->Ellipse(startPoint.x, startPoint.y, newEndPoint.x,
```

```
newEndPoint.y);
            }
            else if (pictureFourStyle == 2)//有边框，白填充
            {
                CPen pen(penStyle, penWide, penColor);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->Ellipse(startPoint.x, startPoint.y, oldEndPoint.x,
oldEndPoint.y);
                pDC->Ellipse(startPoint.x, startPoint.y, newEndPoint.x,
newEndPoint.y);
            }
            else if (pictureFourStyle == 3)//有边框，颜色填充
            {
                CPen pen(penStyle, penWide, penColor);
                CBrush brush(brushColor);
                pDC->SelectObject(&brush);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->Ellipse(startPoint.x, startPoint.y, oldEndPoint.x,
oldEndPoint.y);
                pDC->Ellipse(startPoint.x, startPoint.y, newEndPoint.x,
newEndPoint.y);
            }
            else if (pictureFourStyle == 4)//有边框，线条填充
            {
                brushColor = penColor;
                CPen pen(penStyle, penWide, penColor);
                CBrush brush(brushStyle, brushColor);
                pDC->SelectObject(&brush);
                pDC->SelectObject(&pen);
                pDC->SetROP2(R2_XORPEN);
                pDC->Ellipse(startPoint.x, startPoint.y, oldEndPoint.x,
oldEndPoint.y);
                pDC->Ellipse(startPoint.x, startPoint.y, newEndPoint.x,
newEndPoint.y);
            }
        }
        else if (DrawingType == 6)//实现铅笔功能
        {
            CPen pen(penStyle, penWide, penColor);
            pDC->SelectObject(&pen);
            pDC->MoveTo(oldEndPoint);
            pDC->LineTo(newEndPoint);

            CMFCDrawDoc *pDoc = GetDocument();
            Shapes *shap = new Shapes();
            shap->m_DrawingType = DrawingType;
            shap->m_startPoint = oldEndPoint;
            shap->m_endPoint = newEndPoint;
            shap->m_penStyle = penStyle;
```

```
            shap->m_penWide = penWide;
            shap->m_penColor = penColor;
            pDoc->shapeArray.AddTail(shap);
            pDoc->SetModifiedFlag();
        }
        oldEndPoint = newEndPoint;
    }

    CView::OnMouseMove(nFlags, point);
}
```