C/C++拾遗（二十一）：MFC-简单绘图

2013-08-27 15:06:51

分类： C/C++

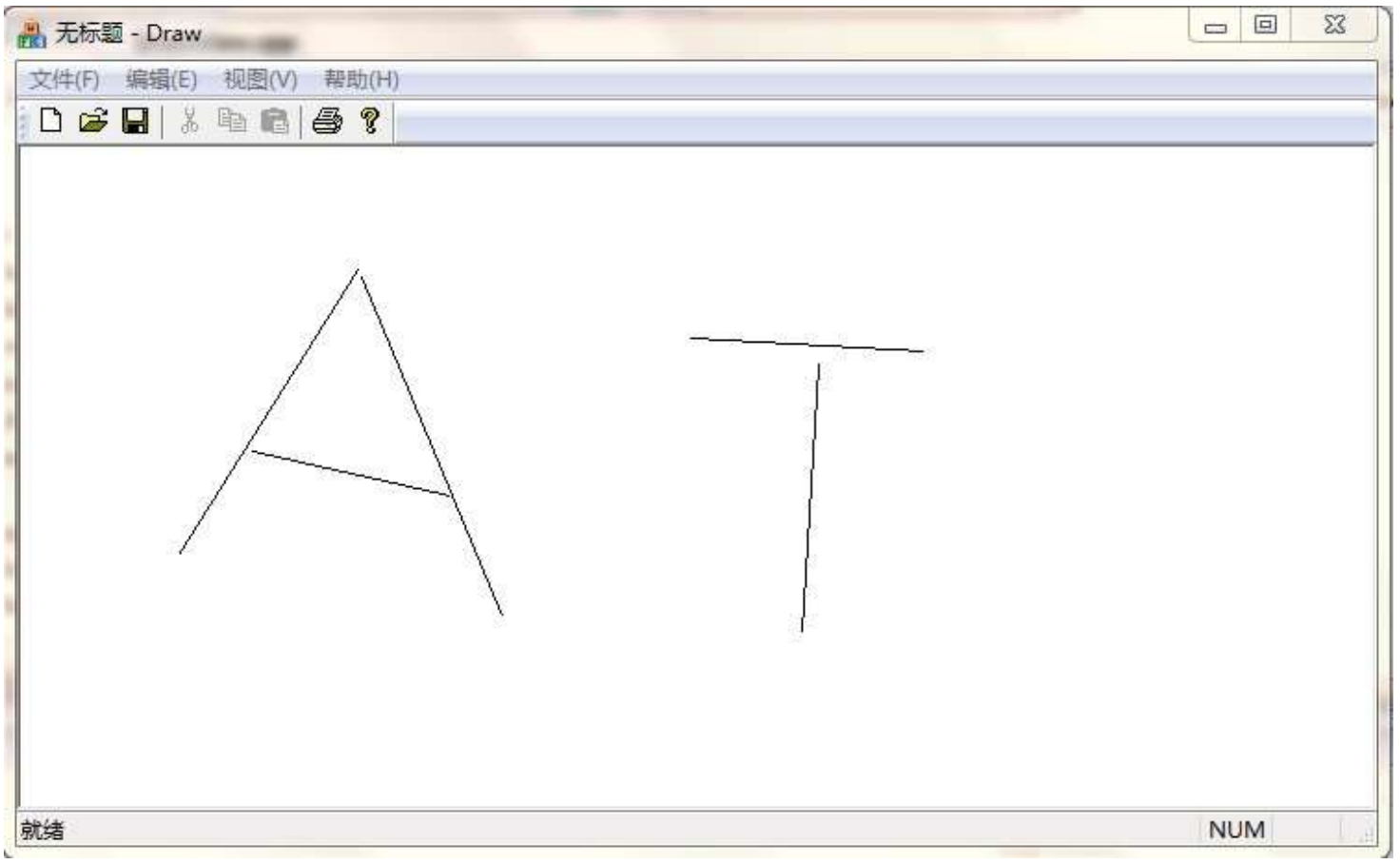
学习MFC就像玩魔术，着实有趣，从传统的黑白命令行跳到熟悉的窗口界面，感觉还是很新鲜的。尽管MFC的知识更多是应用性的，但是自己还是想尽快掌握起来独立地写出图形接口的程序。今天学习的是如何绘制简单的图形，比如直线；当然后面还有高级绘图，但是那些部分对于自己现在就没有多少必要了：自己需要的是GUI，而非全部的MFC。所以接下来自己会挑几个感觉重要的、需要的部分来学习，目标就是看完这些之后自己可以写出一个图形小程序。

言归正传，如果想在窗口上绘制直线的话，我们需要捕捉两个点：线段的起点和终点。二者可以分别通过WM_LBUTTONDOWN和WM_LBUTTONUP来实现。类似于上节课讲述的添加消息函数的方法，在类视图的CDrawView右键属性，选择“消息”菜单，找到对应的消息即可。进入二者的消息函数的实现之前，需要定义一个变量表示线段的起点CPoint m_ptOrigin，同样是右键View类添加变量即可：

点击(此处)折叠或打开

```
1.  void CDrawView::OnLButtonDown(UINT nFlags, CPoint point)
2.  {
3.      // TODO: 在此添加消息处理程序代码和/或调用默认值
4.
5.      //MessageBox(L"View Clicked!");     //用于单击窗口客户区时的测试
6.      m_ptOrigin = point;
7.
8.      CView::OnLButtonDown(nFlags, point);
9.  }
10.
11. void CDrawView::OnLButtonUp(UINT nFlags, CPoint point)
12. {
13.     // TODO: 在此添加消息处理程序代码和/或调用默认值
14. /*
15.     //获得窗口的设备描述表
16.     HDC hdc;
17.     hdc = ::GetDC(m_hWnd);
18.     //移动到线条的起点
19.     MoveToEx(hdc, m_ptOrigin.x, m_ptOrigin.y, NULL);
20.     //画线
21.     LineTo(hdc, point.x, point.y);
22.     //释放设备描述表
23.     ::ReleaseDC(m_hWnd, hdc);
24. */
25.     //使用CDC类实现
26.     CDC *pDC = GetDC();
27.     pDC->MoveTo(m_ptOrigin);
28.     pDC->LineTo(point);
29.     ReleaseDC(pDC);
30.
31.
32.     CView::OnLButtonUp(nFlags, point);
33. }
```

上面的代码使用了设备描述表（DC）来实现绘制直线。MS为程序猿提供了应用程序与设备交互的数据结构DC（Device Context）以实现程序的平台无关性。可以使用直接的DC，也可以使用DC的封装类CDC实现，上面都给出了示例。当然，更加简便地是使用CDC的派生类CClientDC，可以不用自己却写GetDC和ReleaseDC函数。运行测试结果如下：

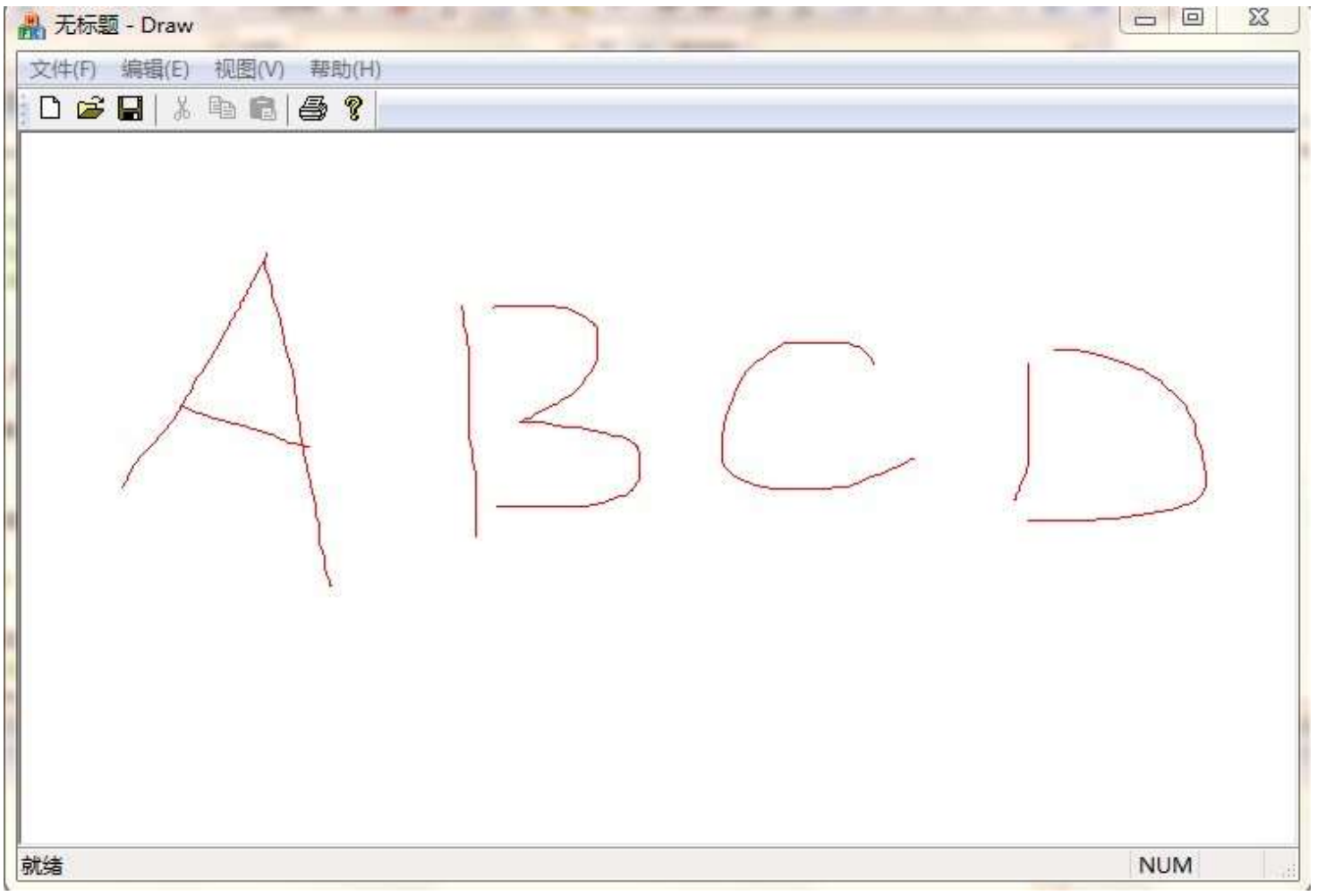


细心的童鞋会发现上面的程序只能绘制直线，那如何绘制连续线呢？其实这里的关键在于实现的想法，如果我们把曲线看作许多小线段的组合，那么每次更新起点连续绘制小线段不就可以了么？但是这里需要注意的是我们需要一个Bool m_bDraw来判断鼠标是否点下是否应当一直绘制。我们需要使用WM_MOUSEMOVE来捕捉鼠标的不断移动：

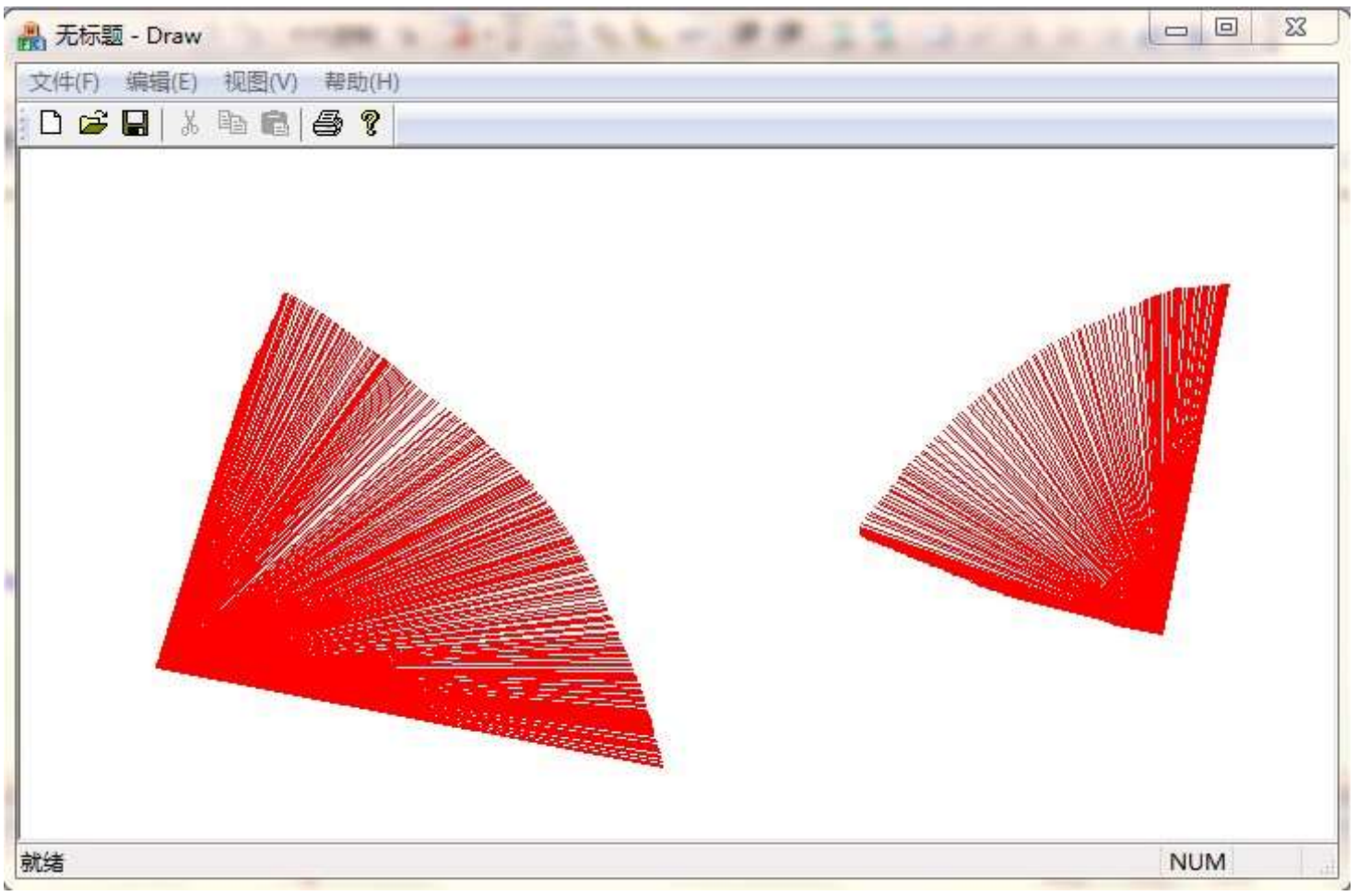
点击(此处)折叠或打开

```
1.  void CDrawView::OnLButtonDown(UINT nFlags, CPoint point)
2.  {
3.      // TODO: 在此添加消息处理程序代码和/或调用默认值
4.
5.      //MessageBox(L"View Clicked!");
6.      m_ptOrigin = point;
7.      m_bDraw = true;
8.
9.      CView::OnLButtonDown(nFlags, point);
10. }
11.
12. void CDrawView::OnLButtonUp(UINT nFlags, CPoint point)
13. {
14.     // TODO: 在此添加消息处理程序代码和/或调用默认值
15.
16.     m_bDraw = false;
17.
18.     CView::OnLButtonUp(nFlags, point);
19. }
20.
21. void CDrawView::OnMouseMove(UINT nFlags, CPoint point)
22. {
23.     // TODO: 在此添加消息处理程序代码和/或调用默认值
24.
25.     CClientDC dc(this);
26.     //创建一个画笔
27.     CPen pen(PS_SOLID, 1, RGB(255, 0,0));
28.     //将创建的画笔选入设备描述表
29.     CPen *pOldPen = dc.SelectObject(&pen);
30.     if (m_bDraw == true)
31.     {
32.         dc.MoveTo(m_ptOrigin);
33.         dc.LineTo(point);
34.         m_ptOrigin = point ;  //每次都更新起点
35.     }
36.
37.
38.     CView::OnMouseMove(nFlags, point);
39. }
```

这样我们就可以绘制曲线了：



其实，只要把上面代码中更新起点的那步去掉，我们就可以得到一个扇形：



神奇吧？是不是也想试试，看到自己的程序出现这样的图形，还是有点小兴奋。不过自己心里明白，自己学习MFC的目的是将其作为一个图形接口工具，根本上的东西还是在于自己研究的对象的编程设计与实现。好了，明确了目标，快马加鞭地继续努力吧！

上一篇：C/C++拾遗（二十）：初识MFC框架
下一篇：VS2008LINK : fatal error LNK1000: Internal error during IncrBuildImage

0