DOI: 10.13657/j.cnki.gxkxyxb.2005.s1.039

基于自定义类移动对象的打印

Printing of the Moved Object Based on User-Defined Class

黄天亮,黄福莹,谢刚强,裴国兴

Huang Tianliang, Huang Fuying, Xie Gangqiang, Pei Guoxing

(广西大学计算机与电子信息学院,广西南宁 530004)

(School of Comp., Elec. and Info., Guangxi Univ., Nanning, Guangxi 530004, China)

摘要:基于自定义类,利用 Visual C++ 语言,给出自定义按钮控件类和自定义移动对象类的打印方法,这两种方法可精确地确定打印位置,获得较好的打印效果。

关键词:移动对象 打印 自定义类

中图法分类号: TP391.9 文献标识码: A 文章编号: 1002-7378(2005)S0-0118-03

Abstract Based on Self-defined class, this paper gives two print methods of the user-defined push-button class and move object class. These methods can exactly orient the printing object, and gain better printing effect.

Key words moved object, printing, user-defined class

一般的打印程序打印所得的内容位置相对固定,但在特殊纸张上打印时存在打印对象不能移动和打印位置不准确等明显缺点。而实际上在一些打印业务需要有专门的打印格式要求,这些要求归纳起来有两点[1]:第一要求打印内容可移动,即用户可以通过鼠标移动打印内容实现灵活排版;第二要求打印内容能准确地打印到指定的位置。为了能满足上述两个要求,本文设计了两个自定义类,利用这两个自定义类能很好地满足用户的打印要求。

1 自定义按钮控件类的打印

在一些需要打印的报表中,有时需要调整打印条目的位置,在打印纸张上已经有了这些条目,只需要指定位置打印出数据[2]。应用下面的自定义按钮控件类能实现这一要求。

1.1 基于 Cbutton 自定义类

自定义类及成员函数如下[3]:

```
class CMyButton: public CButton
{
    private:
      CPoint m_Position;
    public:
      CMyButton();
      virtual ~CMyButton();
    protected:
       afx_msg void OnLButtonUp(UINT nFlags, CPoint point);
       afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
       afx_msg void OnMouseMove(UINT nFlags, CPoint point);
    //nFlags表明了当前一些按键的消息, point表示当前鼠标的设备坐标,坐标原点对应视左上角。
};
```

1.2 主要成员函数注释

```
//选定按钮后,可拖动鼠标实现按钮的移动
void CMyButton::OnMouseMove(UINT nFlags, CPoint point)
{
    //自定义光标变量,将位图 IDC_CURSOR1设为光标图案
    HCURSOR hCursor = AfxGetApp()->LoadCursor(IDC_CURSOR1);
    if(GetCapture()==this)
    {
```

收稿日期: 2005-08-17

作者简介: 黄天亮(1982-),男,广西昭平人,本科生。

```
        RECT rect;
        GetParent()->GetClientRect(&rect);
       ::SetCursor(hCursor);
        ClientToScreen(&point);
        GetParent()->ScreenToClient(&point);
         if(rect.bottom> point.y&&rect.right>
point.x)
     {
     //实时获取当前鼠标设备的句柄
           CDC*  pDC= GetParent()->GetDC
();
           //使用鼠标绘图
           pDC->SetROP2(R2_NOT);
     //移动前的按钮位置
           pDC->MoveTo(m_Position);
           pDC->LineTo(m_Position.x+ 80,m_
Position.y);
           pDC->LineTo(m_Position.x+ 80,m_
Position.y+ 20);
            pDC->LineTo(m_Position.x,m_
Position.y+ 20);
           pDC->LineTo(m_Position);
     //在这里不能使用 pDC->Rectangle(m_
Postion.x,m_Postion.y,m_Postion.x+ len,m_
Postion+ 20);否则拖动时将看不见矩形框内的内
容,出现的是个黑色的矩形框
           m_Position= point;
     //移动后的按钮位置
           pDC->MoveTo(m_Position);
           pDC->LineTo(m_Position.x+ 80,m_
Position.y);
           pDC->LineTo(m_Position.x+ 80,m_
Position.y+ 20);
            pDC->LineTo(m_Position.x,m_
Position.y+ 20);
           pDC->LineTo(m_Position);
           //释放当前缓冲区句柄
           pDC->ReleaseAttribDC();
       }
   }
       else CButton::OnMouseMove(nFlags,
point);
}
```

1.3 创建按钮

```
BOOL CMyButton::Create(LPCTSTR
lpszCaption,DWORD dwStyle,const RECT&
rect,CWnd*  pParentWnd,UINT nID);
```

//lpszCaption是按钮上显示的文字,dwStyle为按钮风格,除了 Windows风格可以使用外(如 WS_CHILD| WS_VISUBLE| WS_BORDER),还有按钮专用的一些风格。rect为窗口所占据的矩形区域,pParentWnd为父窗口指针,nID为该窗口的 ID值,为自己指定。

创建自定义按钮控件的方法移动方便,打印时不会显示出来,在打印报表时能精确定位要打印的内容。

2 自定义移动对象类的打印

当要打印可调整位置的报表或文本时,则可以将报表或文本内容自定义为一个对象类,对鼠标发出的选定,弹起和拖动消息进行响应,就可以实现这个对象类的移动。

2.1 定义移动对象类

```
class CMoveObject
{private:
   CPoint m_Position;  //定义文本框位置坐
标
   LONG   m_Length;  //定义文本框长度
   BOOL   m_Captured;  //定义鼠标的标志
public:
    BOOL CanSelect(Cview*  pView,CPoint
point);  //对选定对象的判断
     BOOL MouseMove(Cview*  pView,
CPoint point);
     BOOL MouseDown(Cview*  pView,
CPoint point);
    BOOL MouseUp(Cview*  pView,CPoint
point);
     //Cview*  pView是视图类指针
     BOOL DrawText(CDC*  pDC,CString
&Text,int ratio);  //在可移动的对象内编辑文本,
使用文本方式输出打印内容。
   CMoveObject();  //初始化对象
   virtual~ CMoveObject();
}
```

2.2 主要成员函数注释

```
//选定对象后,可拖动鼠标实现对象的移动
BOOL CMoveObject::MouseMove(CView*
pView,CPoint point)
{
```

```
int len;
len= m_Length* 10;
//获取当前视图句柄
CDC* pDC= pView-> GetDC();
//选用鼠标绘图
pDC-> SetROP2(R2_NOT);
pDC-> MoveTo(m_Position.x, m_Position.y);
pDC-> LineTo(m_Position.x, m_Position.y + 20);
pDC-> LineTo(m_Position.x+ len, m_Position.y+ 20);
pDC-> LineTo(m_Position.x+ len, m_Position.y);
pDC-> LineTo(m_Position.x, m_Position.y);
m_Position= point;
pDC-> MoveTo(m_Position.x, m_Position.y);
pDC-> LineTo(m_Position.x, m_Position.y + 20);
pDC-> LineTo(m_Position.x+ len, m_Position.y+ 20);
pDC-> LineTo(m_Position.x+ len, m_Position.y);
pDC-> LineTo(m_Position.x, m_Position.y);
pDC-> ReleaseAttribDC();
return TRUE;
}
```

2.3 创建对象

本文无需创建函数,只要在头文件中定义CMoveObject类的对象,即可在程序中使用。自定义移动对象类的方法适用于需要完整打印所见即所得的内容,将内容创建为多个对象,就可以自由编排打印。

以上两种方法在 Visual C++ 6.0中都能成功编译、运行。

3 结束语

本文介绍的两种移动对象的打印方法各有特点,创建按钮控件的方法可移动不需要的打印内容,并精确定位需要的打印部分,而创建移动对象的方法则可以直接移动需要定位打印的内容,方便快捷。将两种方法结合就可以实现任意位置的打印工作。在实践中,我们应用这两种方法在打印证书时获得令人满意的打印效果。

参考文献:

[1] 李鲲程. Visual C++ 打印编程技术与工程实践 [M]. 北京: 人民邮电出版社, 2003.

[2] 韦 源. Visual C++ 6.0自学捷径 [M]. 北京: 北京大学出版社, 1999.

[3] 徐晓刚,高兆法,王秀娟. Visual C++ 6.0入门与提高 [M]. 北京: 清华大学出版社, 1999.

(责任编辑: 黎贞崇)