**vc 实现打印功能**

Visual C++6.0是开发Windows应用程序的强大工具，但是要通过它实现程序的打印功能，一直是初学者的一个难点，经常有朋友询问如何在VC中实现打印功能，他们往往感到在MFC提供的框架内实现这个问题很复杂，不知道如何下手。本例针对这个问题，介绍一种简单的方法实现文字串的打印功能,读者朋友可以在此基础上稍微改动一下，就可以实现文件、图像的打印功能。  
  
　　一、实现方法  
  
　　在Windows操作系统下，显示器、打印机和绘图仪都被视为输出设备，正常情况下，系统默认的输出设备是显示器。要使用打印机,首先需要创建一个指向打印机的设备环境句柄,然后通过该句柄调用相关的绘图函数把所需的文字和图形输出至打印机上。当打印结束后,删除这个设备环境句柄即可。  
  
　　当Windows系统中安装好打印机后，系统总是自动设置一个打印机为系统的默认打印机,在Windows的启动配置文件Win.ini中的[window]段中列出了带有关键字device的默认打印机。下面是某一机器中Win.ini中的[Windows]字段的内容:

|  |
| --- |
| [windows] load= run= NullPort=None device=HP LaserJet 4050(computer000),HPBFDB1,LPT1 |

　　在上述关键字device后的字符串中，包含了系统中默认打印机的三个重要属性，它们依次是打印机的设备名HP LaserJet 4050(computer000)，驱动程序名是HPBFDB1，输出端口为LPT1。  
  
　　为了操纵系统默认的打印机，实现程序的打印功能，在程序中可调用API函数GetProfileString（）从Win.ini文件中获得device这个设备字符串，该函数的原型为：DWORD GetProfileString( LPCTSTR lpAppName, LPCTSTR lpKeyName, LPCTSTR lpDefault, LPTSTR lpReturnedString, DWORD nSize)。函数中lpAppName参数为所要检索的Win.ini文件中的字段名；lpKeyName为字段中的关键字名；lpDefault为默认的字符串；lpReturnedString为检索到的字符串，如果该函数没有从lpKeyName关键字中检索到相应的字符串，则kpRetrunedString返回默认字符串lpDefault；nSize为返回字符串的长度。  
  
　　获取上述字符串后，再使用strtok（）函数将该字符串进行分解,获得与打印机相关的三个属性,作为API函数CreateDC（）创建打印机设备环境句柄的参数，CreateDC（）函数如果调用成功,则为默认打印机创建一个设备环境句柄,否则返回一个空值(NULL)。该函数的原形为：HDC CreateDC(LPCTSTR lpszDriver,LPCTSTR lpszDevice,LPCTSTR lpszOutput,CONST DEVMODE \*lpinitData)。该函数的前三个参数恰好对应打印机的三个属性，最后一个参数为初始化打印机驱动程序的数据，一般情况下该参数设置为NULL就可以了。  
  
　　在具体打印的过程中，调用int StartDoc( HDC hdc, CONST DOCINFO \*lpdi )函数来开始一个打印任务，其中参数lpdi为一个指向DOCINFO结构的指针，该结构如下：

|  |
| --- |
| typedef struct {  　int cbSize; //结构的尺寸大小； 　LPCTSTR lpszDocName; //文档的名字； 　LPCTSTR lpszOutput; //输出文档名，一般情况下为NULL； 　LPCTSTR lpszDatatype;//用来记录打印过程的数据类型，一般情况下为NULL； 　DWORD fwType; //用来支持打印工作的额外 |

信息，一般情况下为NULL；  
} DOCINFO, \*LPDOCINFO;   
　　开始一个打印任务后，再调用StartPage(hdcprint)函数让打印机走纸，通知打印机有文档将要打印；接下来的工作就是输出数据了，这部分工作对于开发人员来说就象往计算机屏幕上输出文字、图像一样容易，只不过是计算机根据当前的设备环境句柄自动将数据输出到打印机罢了。数据打印完后，需要作一些善后处理工作，使用RestoreDC(hdcprint,-1)函数恢复打印机设备句柄、EndPage(hdcprint)函数让打印机停止打印，最后调用EndDoc(hdcprint)函数结束上述的打印作业。  
  
　　二、编程步骤  
  
　　1、启动Visual C++6.0，新建一个基于对话框的应用程序Test，在程序的对话框窗体中加入一个按钮(Button),设置这个Button的属性:ID=IDC\_PRINT,CAPTION="打印"；  
  
　　2、使用Class Wizard类向导为该按钮添加一个鼠标单击处理函数OnPrint（）  
  
　　3、修改TestDlg.cpp文件中的OnPrint（）函数；  
  
　　4、添加代码，编译运行程序。

==========================================================================================================================

在打印预览对话框类中

**[cpp]**  [view plain](http://blog.csdn.net/xgx198831/article/details/7024949)  [copy](http://blog.csdn.net/xgx198831/article/details/7024949)

**extern** **int** nPaperSize\_X ;

**extern** **int** nPaperSize\_Y ;

**extern** **int** nOneLines;

**extern** **int** nNextLines;

//打印结构

**typedef** **struct**

{

**int**  nMaxLine;   //最大行数

**int**  nCountPage;   //一共页数

**int**  nCurPage;   //当前页码

**BOOL** IsPrint;   //是否打印

**HWND** hWnd;    //窗口句柄

**HWND** hListView;   //列表控件句柄

**TCHAR** szTag[256];   //其它数据

**int**  nTag;    //其它数据

**LPVOID** lpVoid;    //其它数据

 CGridCtrlEx \*pObj;   //区分是月报表还是日报表

}PRNINFO, \*PPRNINFO;

**[cpp]**  [view plain](http://blog.csdn.net/xgx198831/article/details/7024949)  [copy](http://blog.csdn.net/xgx198831/article/details/7024949)

//回调函数，设置打印属性

**void** CPreviewParentDlg::SetCallBackFun( PRINTPREVIEW pFun, PRNINFO &sPrnInfo )

{

 memcpy(&m\_PrnInfo, &sPrnInfo, **sizeof**(PRNINFO));

 m\_pDrawInfoFun = pFun;

 m\_nCount = m\_PrnInfo.nMaxLine;  // 总的行数

 m\_nCountPage = 1;

**int** m = m\_nCount-m\_OneCount;

**int** n = m/m\_NextCount;

 m\_nCountPage += n;

 n = m%m\_NextCount;

**if**(n>0)

  m\_nCountPage++;   // 页数

 m\_PrnInfo.nCountPage = m\_nCountPage;

 sPrnInfo.nCountPage = m\_nCountPage;

}

**[cpp]**  [view plain](http://blog.csdn.net/xgx198831/article/details/7024949)  [copy](http://blog.csdn.net/xgx198831/article/details/7024949)

**void** CPreviewChildDlg::PrintDoc()

{

    NotifyDlg Ndlg(\_T("决定打印当前报表吗?"), TRUE);

**if** (Ndlg.DoModal() == IDCANCEL)

**return**;

    PRINTDLG printInfo;

    ZeroMemory(&printInfo,**sizeof**(printInfo));  //清空该结构

    printInfo.lStructSize = **sizeof**(printInfo);

    printInfo.hwndOwner = 0;

    printInfo.hDevMode = 0;

    printInfo.hDevNames = 0;

    //这个是关键，PD\_RETURNDC 如果不设这个标志，就拿不到hDC了

    //            PD\_RETURNDEFAULT 这个就是得到默认打印机，不需要弹设置对话框

    printInfo.Flags = PD\_RETURNDC | PD\_RETURNDEFAULT | PD\_ALLPAGES;

    PrintDlg(&printInfo);//调用API拿出默认打印机

**DWORD** rst = CommDlgExtendedError();//看看出错没有

**if**(rst != 0)

    {//出错了，清空标志再次调用API，此时就会弹出打印设置对话框供用户选择了

        printInfo.Flags = 0;

        PrintDlg(&printInfo);

    }

**HDC** printDC=printInfo.hDC; //得到打印DC，输出到打印，

    CDC MemDc;

    MemDc.Attach(printDC);

**if**(m\_pDrawInfoFun!= NULL)

    {

        m\_PrnInfo.IsPrint = TRUE;  // 用打印机打印

        m\_PrnInfo.nCurPage = m\_CurPage;

        m\_PrnInfo.nCountPage = m\_CountPage;

        m\_pDrawInfoFun(MemDc, m\_PrnInfo);

    }

    MemDc.DeleteDC();}

**[cpp]**  [view plain](http://blog.csdn.net/xgx198831/article/details/7024949)  [copy](http://blog.csdn.net/xgx198831/article/details/7024949)

//刷新预览区

**void** CPreviewChildDlg::OnPaint()

{

    CPaintDC dc(**this**); // device context for painting

    // TODO: Add your message handler code here

    CClientDC dlgDC(**this**);

    SetWindowOrgEx(dlgDC.m\_hDC, m\_xPt, m\_yPt, NULL);

    CDC MemDc;

    MemDc.CreateCompatibleDC(NULL);

    CBitmap cBitmap;

**int** xP = dlgDC.GetDeviceCaps(LOGPIXELSX);

**int** yP = dlgDC.GetDeviceCaps(LOGPIXELSY);

    DOUBLE xPix = (DOUBLE)xP\*10/254;    //每 mm 宽度的像素

    DOUBLE yPix = (DOUBLE)yP\*10/254;    //每 mm 高度的像素

    cBitmap.CreateCompatibleBitmap(&dlgDC, nPaperSize\_X\*xPix, nPaperSize\_Y\*yPix);

    MemDc.SelectObject(&cBitmap);

**if**(m\_pDrawInfoFun!= NULL)

    {

        m\_PrnInfo.IsPrint = FALSE;  //显示的是 预览窗口

        m\_PrnInfo.nCurPage = m\_CurPage;

        m\_pDrawInfoFun(MemDc, m\_PrnInfo);   //调用回调函数

    }

    dlgDC.BitBlt(xP/2, yP/2, nPaperSize\_X\*xPix+xP/2, nPaperSize\_Y\*yPix+yP/2, &MemDc, 0, 0, SRCCOPY);

    MemDc.DeleteDC();

    cBitmap.DeleteObject();

    // Do not call CDialog::OnPaint() for painting messages

}

=============调用打印功能类

**[cpp]**  [view plain](http://blog.csdn.net/xgx198831/article/details/7024949)  [copy](http://blog.csdn.net/xgx198831/article/details/7024949)

**void** CAttendReportDlg::PrintData()

{

    CGridCtrlEx \*pGridCtrl = NULL;

**BOOL** bDay = FALSE;

**if** ( ((CButton\*)GetDlgItem(IDC\_RADIO\_DAY))->GetCheck() )

    {

        pGridCtrl = m\_pDayGridCtrl;

        bDay = TRUE;

    }

**else** **if** ( ((CButton\*)GetDlgItem(IDC\_RADIO\_MONTH))->GetCheck() )

    {

        pGridCtrl = m\_pMonGridCtrl;

    }

**if**( pGridCtrl->GetRowCount() <= 1 )    // 没有记录

**return**;

    ///选择打印机对话框

    CDC MemDc;

**HDC** hdcPrint = NULL;

    CPrintDialog dlg(FALSE);

**if** (m\_bPrint)  //打印按钮，不弹出选择对话框，获取默认打印设备

    {

        PRINTDLG printInfo;

        ZeroMemory(&printInfo,**sizeof**(printInfo));  //清空该结构

        printInfo.lStructSize = **sizeof**(printInfo);

        printInfo.hwndOwner = 0;

        printInfo.hDevMode = 0;

        printInfo.hDevNames = 0;

        //这个是关键，PD\_RETURNDC 如果不设这个标志，就拿不到hDC了

        //PD\_RETURNDEFAULT 这个就是得到默认打印机，不需要弹出设置对话框

        printInfo.Flags = PD\_RETURNDC | PD\_RETURNDEFAULT | PD\_ALLPAGES;

        PrintDlg(&printInfo);//调用API拿出默认打印机

**DWORD** rst = CommDlgExtendedError();//看看出错没有

**if**(rst != 0)

        {//出错了，清空标志再次调用API，此时就会弹出打印设置对话框供用户选择了

            printInfo.Flags = 0;

            PrintDlg(&printInfo);

        }

        hdcPrint=printInfo.hDC; //得到打印DC，输出到打印

    }

**else**  //弹出对话框选择打印设备

    {

        dlg.DoModal();

        hdcPrint = dlg.GetPrinterDC();

    }

**if**(hdcPrint == NULL)

    {

        NotifyDlg Ndlg(\_T("打印机初始化失败!"));

        Ndlg.DoModal();

**return**;

    }

    MemDc.Attach(hdcPrint);

    nPaperSize\_X = MemDc.GetDeviceCaps(HORZSIZE);    // 纸张宽度

    nPaperSize\_Y = MemDc.GetDeviceCaps(VERTSIZE);    // 纸张高度

**int** xP = GetDeviceCaps(MemDc.m\_hDC, LOGPIXELSX);    //x方向每英寸像素点数

**int** yP = GetDeviceCaps(MemDc.m\_hDC, LOGPIXELSY);    //y方向每英寸像素点数

**int** xPix = (DOUBLE)xP\*10/254;   //每 mm 宽度的像素

**int** yPix = (DOUBLE)yP\*10/254;   //每 mm 高度的像素

    DOUBLE fAdd = 5\*yPix;       //每格递增量

    nOneLines = (nPaperSize\_Y \* 0.85\*yPix)/fAdd;

    nNextLines = (nPaperSize\_Y \* 0.85\*yPix)/fAdd+1;

    PRNINFO PrnInfo = {0};

    PrnInfo.hListView = NULL;

    PrnInfo.hWnd = **this**->m\_hWnd;

    PrnInfo.IsPrint = m\_bPrint;

    PrnInfo.nCurPage = 1;

    PrnInfo.nMaxLine = pGridCtrl->GetRowCount()-1;

    PrnInfo.pObj = pGridCtrl;

    CPreviewParentDlg DlgPreView;

    CPreviewChildDlg DlgChildPreView;

**if** (bDay)

    {

        DlgPreView.SetCallBackFun(PrintDayInfo, PrnInfo);  //回调函数，设置打印或预览函数，及纸张排版信息

        DlgChildPreView.SetCallBackFun(PrintDayInfo, PrnInfo);

    }

**else**

    {

        DlgPreView.SetCallBackFun(PrintMonInfo, PrnInfo);

        DlgChildPreView.SetCallBackFun(PrintMonInfo, PrnInfo);

    }

**if** (!m\_bPrint)

    {

        DlgPreView.DoModal();

    }

**else**

    {

        DlgChildPreView.PrintDoc();

    }

    MemDc.DeleteDC();

}

**void** CAttendReportDlg::PrintDayInfo( CDC &memDC, PRNINFO PrnInfo )

{

**if**(memDC.m\_hDC == NULL)

**return**;

**int** nCurPage = PrnInfo.nCurPage;    //当前页

**BOOL** IsPrint = PrnInfo.IsPrint;     //是否打印

**int** nMaxPage = PrnInfo.nCountPage;  //最大页码

**HWND** hWnd = PrnInfo.hWnd;

    CString csLFinality, csRFinality;

    CGridCtrlEx \*pGridCtrl = PrnInfo.pObj;

    CTime time = CTime::GetCurrentTime();

    csLFinality = time.Format(\_T("%Y-%m-%d %H:%M:%S"));

    csLFinality = \_T("报表日期:") + csLFinality;

    csRFinality.Format(\_T("第 %i 页/共 %i 页"), nCurPage, nMaxPage);

**TCHAR** szTitle[] = \_T("考 勤 日 报 表");

    CRect rc, rt1, rt2, rt3, rt4, rt5, rt6, rt7, rt8, rt9, rt10;

    CPen \*hPenOld;

    CPen cPen;

    CFont TitleFont, DetailFont, \*oldfont;

    //标题字体

    TitleFont.CreateFont(-MulDiv(14,memDC.GetDeviceCaps(LOGPIXELSY),72),

        0,0,0,FW\_NORMAL,0,0,0,GB2312\_CHARSET,

        OUT\_STROKE\_PRECIS,CLIP\_STROKE\_PRECIS,DRAFT\_QUALITY,

        VARIABLE\_PITCH|FF\_SWISS,\_T("黑体"));

    //细节字体

    DetailFont.CreateFont(-MulDiv(10,memDC.GetDeviceCaps(LOGPIXELSY),92),

        0,0,0,FW\_NORMAL,0,0,0,GB2312\_CHARSET,

        OUT\_STROKE\_PRECIS,CLIP\_STROKE\_PRECIS,DRAFT\_QUALITY,

        VARIABLE\_PITCH|FF\_SWISS,\_T("宋体"));

    //粗笔

    cPen.CreatePen(PS\_SOLID, 2, RGB(0, 0, 0));

**int** xP = GetDeviceCaps(memDC.m\_hDC, LOGPIXELSX);    //x方向每英寸像素点数

**int** yP = GetDeviceCaps(memDC.m\_hDC, LOGPIXELSY);    //y方向每英寸像素点数

    DOUBLE xPix = (DOUBLE)xP\*10/254;    //每 mm 宽度的像素

    DOUBLE yPix = (DOUBLE)yP\*10/254;    //每 mm 高度的像素

    DOUBLE fAdd = 5\*yPix;       //每格递增量

    DOUBLE nTop = 30\*yPix;      //第一页最上线

**int**   iStart = 0;           //从第几行开始读取

    DOUBLE nBottom = nTop+nOneLines\*fAdd;

**if**(nCurPage != 1)

        nTop = 30\*yPix-fAdd;    //非第一页最上线

**if**(nCurPage == 2)

        iStart = nOneLines;

**if**(nCurPage>2)

        iStart = nOneLines+(nCurPage - 2)\*nNextLines;

    DOUBLE nLeft = 15\*xPix;         //最左线

    DOUBLE nRight = xPix\*(nPaperSize\_X-15); //最右线

    DOUBLE nItemWide = ((nPaperSize\_X-30)/14)\*xPix;

    DOUBLE nTextAdd = 1.5\*xPix;

**if**(IsPrint)

    {

        //真正打印部分

**static** DOCINFO di = {**sizeof** (DOCINFO),  szTitle} ;

        //开始文档打印/     start print

         //

**if**(memDC.StartDoc( &di ) < 0) // startdoc-----enddoc

        {

            NotifyDlg dlg(\_T("连接到打印机化败!"));

            dlg.DoModal();

        }

**else**

        {

            iStart = 0;

            nTop = 30\*yPix;     //第一页最上线

**for**(**int** iTotalPages = 1; iTotalPages<=nMaxPage; iTotalPages++)

            {

**int** nCurPage = iTotalPages;

                csRFinality.Format(\_T("第 %i 页/共 %i 页"), nCurPage, nMaxPage);

                time=CTime::GetCurrentTime();

                csLFinality = time.Format(\_T("%Y-%m-%d %H:%M:%S"));

                csLFinality = \_T("报表日期:") + csLFinality;

**if**(nCurPage != 1)

                    nTop = 30\*yPix-fAdd;    //非第一页最上线

**if**(nCurPage == 2)

                    iStart = nOneLines;

**if**(nCurPage>2)

                    iStart = nOneLines+(nCurPage - 2)\*nNextLines;

                //开始页

**if**(memDC.StartPage() < 0)

                {

                    NotifyDlg dlg(\_T("打印失败!"));

                    dlg.DoModal();

                    memDC.AbortDoc();

**return**;

                }

**else**

                {

                    //打印

                    //标题

                    oldfont = memDC.SelectObject(&TitleFont);

**int** nItem = nNextLines;

**if**(nCurPage == 1)

                    {

                        nItem = nOneLines;

                        rc.SetRect(0, yPix\*15, nPaperSize\_X\*xPix, yPix\*25);

                        memDC.DrawText(szTitle, &rc, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    }

                    //细节

                    memDC.SelectObject(&DetailFont);

                    rc.SetRect(nLeft, nTop, nRight, nTop+fAdd);

                    //上横线

                    memDC.MoveTo(rc.left, rc.top);

                    memDC.LineTo(rc.right, rc.top);

                    rt1.SetRect(nLeft, nTop, rc.right -12.4\*nItemWide , nTop+fAdd);     //编 号

                    rt2.SetRect(rt1.right, rt1.top, rt1.right + 1.5\*nItemWide, rt1.bottom); //姓名

                    rt3.SetRect(rt2.right, rt1.top, rt2.right + 1.5\*nItemWide, rt1.bottom); //考勤日期

                    rt4.SetRect(rt3.right, rt1.top, rt3.right + 2.2\*nItemWide, rt1.bottom); //班次

                    rt5.SetRect(rt4.right, rt1.top, rt4.right + 1.6\*nItemWide, rt1.bottom); //时段

                    rt6.SetRect(rt5.right, rt1.top, rt5.right + 1.6\*nItemWide, rt1.bottom); //考勤时间

                    rt7.SetRect(rt6.right, rt1.top, rt6.right + nItemWide, rt1.bottom); //迟到(分)

                    rt8.SetRect(rt7.right, rt1.top, rt7.right + nItemWide, rt1.bottom); //早退(分)

                    rt9.SetRect(rt8.right, rt1.top, rt8.right + nItemWide, rt1.bottom); //旷工(分)

                    rt10.SetRect(rt9.right, rt1.top, rc.right, rt1.bottom); //请假(分)

                    memDC.DrawText(\_T("编 号"), &rt1, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.DrawText(\_T("姓 名"), &rt2, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.DrawText(\_T("考勤日期"), &rt3, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.DrawText(\_T("班 次"), &rt4, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.DrawText(\_T("时 段"), &rt5, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.DrawText(\_T("考勤时间"), &rt6, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.DrawText(\_T("迟到(分)"), &rt7, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.DrawText(\_T("早退(分)"), &rt8, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.DrawText(\_T("旷工(分)"), &rt9, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.DrawText(\_T("请假(分)"), &rt10, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.MoveTo(rt1.right, rt1.top);

                    memDC.LineTo(rt1.right, rt1.bottom);

                    memDC.MoveTo(rt2.right, rt1.top);

                    memDC.LineTo(rt2.right, rt1.bottom);

                    memDC.MoveTo(rt3.right, rt1.top);

                    memDC.LineTo(rt3.right, rt1.bottom);

                    memDC.MoveTo(rt4.right, rt1.top);

                    memDC.LineTo(rt4.right, rt1.bottom);

                    memDC.MoveTo(rt5.right, rt1.top);

                    memDC.LineTo(rt5.right, rt1.bottom);

                    memDC.MoveTo(rt6.right, rt1.top);

                    memDC.LineTo(rt6.right, rt1.bottom);

                    memDC.MoveTo(rt7.right, rt1.top);

                    memDC.LineTo(rt7.right, rt1.bottom);

                    memDC.MoveTo(rt8.right, rt1.top);

                    memDC.LineTo(rt8.right, rt1.bottom);

                    memDC.MoveTo(rt9.right, rt1.top);

                    memDC.LineTo(rt9.right, rt1.bottom);

                    memDC.MoveTo(rc.left, rt1.bottom);

                    memDC.LineTo(rc.right, rt1.bottom);

                    CString strID, strName, strDate, strSID, strTime, strAttTime, strLate, strEarlier, strAbsent, strLeave;

                    rc.SetRect(nLeft, nTop+fAdd, nRight, nTop+2\*fAdd);

                    rt1.SetRect(nLeft+nTextAdd, rc.top, rc.right-12.4\*nItemWide, rc.bottom);

                    rt2.SetRect(rt1.right+nTextAdd, rt1.top, rt1.right + 1.5\*nItemWide, rt1.bottom);

                    rt3.SetRect(rt2.right+nTextAdd, rt1.top, rt2.right + 1.5\*nItemWide, rt1.bottom);

                    rt4.SetRect(rt3.right+nTextAdd, rt1.top, rt3.right + 2.2\*nItemWide, rt1.bottom);

                    rt5.SetRect(rt4.right+nTextAdd, rt1.top, rt4.right +1.6\*nItemWide, rt1.bottom);

                    rt6.SetRect(rt5.right+nTextAdd, rt1.top, rt5.right + 1.6\*nItemWide, rt1.bottom);

                    rt7.SetRect(rt6.right+nTextAdd, rt1.top, rt6.right + nItemWide, rt1.bottom);

                    rt8.SetRect(rt7.right+nTextAdd, rt1.top, rt7.right + nItemWide, rt1.bottom);

                    rt9.SetRect(rt8.right+nTextAdd, rt1.top, rt8.right + nItemWide, rt1.bottom);

                    rt10.SetRect(rt9.right+nTextAdd, rt1.top, rc.right, rt1.bottom);

**int** nCountItem = pGridCtrl->GetRowCount();

**for**(**int** i=1;i<nItem; i++)

                    {

                        strID = pGridCtrl->GetItemText(i+iStart, 1);

                        strName = pGridCtrl->GetItemText(i+iStart, 2);

                        strDate = pGridCtrl->GetItemText(i+iStart, 3);

                        strSID = pGridCtrl->GetItemText(i+iStart, 4);

                        strTime = pGridCtrl->GetItemText(i+iStart, 5);

                        strAttTime = pGridCtrl->GetItemText(i+iStart, 6);

                        strLate = pGridCtrl->GetItemText(i+iStart, 7);

                        strEarlier = pGridCtrl->GetItemText(i+iStart, 8);

                        strAbsent = pGridCtrl->GetItemText(i+iStart, 9);

                        strLeave = pGridCtrl->GetItemText(i+iStart, 10);

                        memDC.DrawText(strID, &rt1, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                        memDC.DrawText(strName, &rt2, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                        memDC.DrawText(strDate, &rt3, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                        memDC.DrawText(strSID, &rt4, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                        memDC.DrawText(strTime, &rt5, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                        memDC.DrawText(strAttTime, &rt6, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                        memDC.DrawText(strLate, &rt7, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                        memDC.DrawText(strEarlier, &rt8, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                        memDC.DrawText(strAbsent, &rt9, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                        memDC.DrawText(strLeave, &rt10, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

                        //下横线

                        memDC.MoveTo(rc.left, rc.bottom);

                        memDC.LineTo(rc.right, rc.bottom);

                        memDC.MoveTo(rt1.right, rt1.top);

                        memDC.LineTo(rt1.right, rt1.bottom);

                        memDC.MoveTo(rt2.right, rt1.top);

                        memDC.LineTo(rt2.right, rt1.bottom);

                        memDC.MoveTo(rt3.right, rt1.top);

                        memDC.LineTo(rt3.right, rt1.bottom);

                        memDC.MoveTo(rt4.right, rt1.top);

                        memDC.LineTo(rt4.right, rt1.bottom);

                        memDC.MoveTo(rt5.right, rt1.top);

                        memDC.LineTo(rt5.right, rt1.bottom);

                        memDC.MoveTo(rt6.right, rt1.top);

                        memDC.LineTo(rt6.right, rt1.bottom);

                        memDC.MoveTo(rt7.right, rt1.top);

                        memDC.LineTo(rt7.right, rt1.bottom);

                        memDC.MoveTo(rt8.right, rt1.top);

                        memDC.LineTo(rt8.right, rt1.bottom);

                        memDC.MoveTo(rt9.right, rt1.top);

                        memDC.LineTo(rt9.right, rt1.bottom);

                        memDC.MoveTo(rc.left, rt1.bottom);

                        memDC.LineTo(rc.right, rt1.bottom);

                        rc.top += fAdd;

                        rc.bottom += fAdd;

                        rt1.top = rc.top;

                        rt1.bottom = rc.bottom;

                        rt2.top = rt1.top;

                        rt2.bottom = rt1.bottom;

                        rt3.top = rt1.top;

                        rt3.bottom = rt1.bottom;

                        rt4.top = rt1.top;

                        rt4.bottom = rt1.bottom;

                        rt5.top = rt1.top;

                        rt5.bottom = rt1.bottom;

                        rt6.top = rt1.top;

                        rt6.bottom = rt1.bottom;

                        rt7.top = rt1.top;

                        rt7.bottom = rt1.bottom;

                        rt8.top = rt1.top;

                        rt8.bottom = rt1.bottom;

                        rt9.top = rt1.top;

                        rt9.bottom = rt1.bottom;

                        rt10.top = rt1.top;

                        rt10.bottom = rt1.bottom;

**if**((i+iStart+1)>=nCountItem)

**break**;

                    }

                    //结尾

                    memDC.MoveTo(rc.left, nTop);

                    memDC.LineTo(rc.left, rc.top);

                    memDC.MoveTo(rc.right, nTop);

                    memDC.LineTo(rc.right, rc.top);

                    memDC.DrawText(csLFinality, &rc, DT\_LEFT | DT\_VCENTER | DT\_SINGLELINE);

                    memDC.DrawText(csRFinality, &rc, DT\_RIGHT| DT\_VCENTER | DT\_SINGLELINE);

                    memDC.EndPage();

                    memDC.SelectObject(oldfont);

                }

            }

            memDC.EndDoc();

        }

    }

**else**

    {

        打印预览

        //边框线

        hPenOld = memDC.SelectObject(&cPen);

        rc.SetRect(0, 0, nPaperSize\_X\*xPix, nPaperSize\_Y\*yPix);

        memDC.Rectangle(&rc);

        memDC.SelectObject(hPenOld);

        //标题

        oldfont = memDC.SelectObject(&TitleFont);

**int** nItem = nNextLines;

**if**(nCurPage == 1)

        {

            nItem = nOneLines;

            rc.SetRect(0, yPix\*15, nPaperSize\_X\*xPix, yPix\*25);

            memDC.DrawText(szTitle, &rc, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        }

        //细节

        memDC.SelectObject(&DetailFont);

        rc.SetRect(nLeft, nTop, nRight, nTop+fAdd);

        //上横线

        memDC.MoveTo(rc.left, rc.top);

        memDC.LineTo(rc.right, rc.top);

        rt1.SetRect(nLeft, nTop, rc.right -12.2\*nItemWide, nTop+fAdd);      //编 号

        rt2.SetRect(rt1.right, rt1.top, rt1.right + 1.5\*nItemWide, rt1.bottom); //姓名

        rt3.SetRect(rt2.right, rt1.top, rt2.right + 1.5\*nItemWide, rt1.bottom); //考勤日期

        rt4.SetRect(rt3.right, rt1.top, rt3.right + 2\*nItemWide, rt1.bottom);   //班次

        rt5.SetRect(rt4.right, rt1.top, rt4.right + 1.6\*nItemWide, rt1.bottom); //时段

        rt6.SetRect(rt5.right, rt1.top, rt5.right + 1.6\*nItemWide, rt1.bottom); //考勤时间

        rt7.SetRect(rt6.right, rt1.top, rt6.right + nItemWide, rt1.bottom); //迟到(分)

        rt8.SetRect(rt7.right, rt1.top, rt7.right + nItemWide, rt1.bottom); //早退(分)

        rt9.SetRect(rt8.right, rt1.top, rt8.right + nItemWide, rt1.bottom); //旷工(分)

        rt10.SetRect(rt9.right, rt1.top, rc.right, rt1.bottom); //请假(分)

        memDC.DrawText(\_T("编 号"), &rt1, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        memDC.DrawText(\_T("姓 名"), &rt2, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        memDC.DrawText(\_T("考勤日期"), &rt3, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        memDC.DrawText(\_T("班 次"), &rt4, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        memDC.DrawText(\_T("时 段"), &rt5, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        memDC.DrawText(\_T("考勤时间"), &rt6, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        memDC.DrawText(\_T("迟到(分)"), &rt7, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        memDC.DrawText(\_T("早退(分)"), &rt8, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        memDC.DrawText(\_T("旷工(分)"), &rt9, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        memDC.DrawText(\_T("请假(分)"), &rt10, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

        memDC.MoveTo(rt1.right, rt1.top);

        memDC.LineTo(rt1.right, rt1.bottom);

        memDC.MoveTo(rt2.right, rt1.top);

        memDC.LineTo(rt2.right, rt1.bottom);

        memDC.MoveTo(rt3.right, rt1.top);

        memDC.LineTo(rt3.right, rt1.bottom);

        memDC.MoveTo(rt4.right, rt1.top);

        memDC.LineTo(rt4.right, rt1.bottom);

        memDC.MoveTo(rt5.right, rt1.top);

        memDC.LineTo(rt5.right, rt1.bottom);

        memDC.MoveTo(rt6.right, rt1.top);

        memDC.LineTo(rt6.right, rt1.bottom);

        memDC.MoveTo(rt7.right, rt1.top);

        memDC.LineTo(rt7.right, rt1.bottom);

        memDC.MoveTo(rt8.right, rt1.top);

        memDC.LineTo(rt8.right, rt1.bottom);

        memDC.MoveTo(rt9.right, rt1.top);

        memDC.LineTo(rt9.right, rt1.bottom);

        memDC.MoveTo(rc.left, rt1.bottom);

        memDC.LineTo(rc.right, rt1.bottom);

        CString strID, strName, strDate, strSID, strTime, strAttTime, strLate, strEarlier, strAbsent, strLeave;

        rc.SetRect(nLeft, nTop+fAdd, nRight, nTop+2\*fAdd);

        rt1.SetRect(nLeft+nTextAdd, rc.top, rc.right-12.2\*nItemWide, rc.bottom);

        rt2.SetRect(rt1.right+nTextAdd, rt1.top, rt1.right + 1.5\*nItemWide, rt1.bottom);

        rt3.SetRect(rt2.right+nTextAdd, rt1.top, rt2.right + 1.5\*nItemWide, rt1.bottom);

        rt4.SetRect(rt3.right+nTextAdd, rt1.top, rt3.right + 2\*nItemWide, rt1.bottom);

        rt5.SetRect(rt4.right+nTextAdd, rt1.top, rt4.right + 1.6\*nItemWide, rt1.bottom);

        rt6.SetRect(rt5.right+nTextAdd, rt1.top, rt5.right + 1.6\*nItemWide, rt1.bottom);

        rt7.SetRect(rt6.right+nTextAdd, rt1.top, rt6.right + nItemWide, rt1.bottom);

        rt8.SetRect(rt7.right+nTextAdd, rt1.top, rt7.right + nItemWide, rt1.bottom);

        rt9.SetRect(rt8.right+nTextAdd, rt1.top, rt8.right + nItemWide, rt1.bottom);

        rt10.SetRect(rt9.right+nTextAdd, rt1.top, rc.right, rt1.bottom);

**int** nCountItem = pGridCtrl->GetRowCount();

**for**(**int** i=1;i<nItem; i++)

        {

            strID = pGridCtrl->GetItemText(i+iStart, 1);

            strName = pGridCtrl->GetItemText(i+iStart, 2);

            strDate = pGridCtrl->GetItemText(i+iStart, 3);

            strSID = pGridCtrl->GetItemText(i+iStart, 4);

            strTime = pGridCtrl->GetItemText(i+iStart, 5);

            strAttTime = pGridCtrl->GetItemText(i+iStart, 6);

            strLate = pGridCtrl->GetItemText(i+iStart, 7);

            strEarlier = pGridCtrl->GetItemText(i+iStart, 8);

            strAbsent = pGridCtrl->GetItemText(i+iStart, 9);

            strLeave = pGridCtrl->GetItemText(i+iStart, 10);

            memDC.DrawText(strID, &rt1, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

            memDC.DrawText(strName, &rt2, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

            memDC.DrawText(strDate, &rt3, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

            memDC.DrawText(strSID, &rt4, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

            memDC.DrawText(strTime, &rt5, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

            memDC.DrawText(strAttTime, &rt6, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

            memDC.DrawText(strLate, &rt7, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

            memDC.DrawText(strEarlier, &rt8, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

            memDC.DrawText(strAbsent, &rt9, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

            memDC.DrawText(strLeave, &rt10, DT\_CENTER | DT\_VCENTER | DT\_SINGLELINE);

            //下横线

            memDC.MoveTo(rc.left, rc.bottom);

            memDC.LineTo(rc.right, rc.bottom);

            memDC.MoveTo(rt1.right, rt1.top);

            memDC.LineTo(rt1.right, rt1.bottom);

            memDC.MoveTo(rt2.right, rt1.top);

            memDC.LineTo(rt2.right, rt1.bottom);

            memDC.MoveTo(rt3.right, rt1.top);

            memDC.LineTo(rt3.right, rt1.bottom);

            memDC.MoveTo(rt4.right, rt1.top);

            memDC.LineTo(rt4.right, rt1.bottom);

            memDC.MoveTo(rt5.right, rt1.top);

            memDC.LineTo(rt5.right, rt1.bottom);

            memDC.MoveTo(rt6.right, rt1.top);

            memDC.LineTo(rt6.right, rt1.bottom);

            memDC.MoveTo(rt7.right, rt1.top);

            memDC.LineTo(rt7.right, rt1.bottom);

            memDC.MoveTo(rt8.right, rt1.top);

            memDC.LineTo(rt8.right, rt1.bottom);

            memDC.MoveTo(rt9.right, rt1.top);

            memDC.LineTo(rt9.right, rt1.bottom);

            memDC.MoveTo(rc.left, rt1.bottom);

            memDC.LineTo(rc.right, rt1.bottom);

            rc.top += fAdd;

            rc.bottom += fAdd;

            rt1.top = rc.top;

            rt1.bottom = rc.bottom;

            rt2.top = rt1.top;

            rt2.bottom = rt1.bottom;

            rt3.top = rt1.top;

            rt3.bottom = rt1.bottom;

            rt4.top = rt1.top;

            rt4.bottom = rt1.bottom;

            rt5.top = rt1.top;

            rt5.bottom = rt1.bottom;

            rt6.top = rt1.top;

            rt6.bottom = rt1.bottom;

            rt7.top = rt1.top;

            rt7.bottom = rt1.bottom;

            rt8.top = rt1.top;

            rt8.bottom = rt1.bottom;

            rt9.top = rt1.top;

            rt9.bottom = rt1.bottom;

            rt10.top = rt1.top;

            rt10.bottom = rt1.bottom;

**if**((i+iStart+1)>=nCountItem)

**break**;

        }

        //结尾

        memDC.MoveTo(rc.left, nTop);

        memDC.LineTo(rc.left, rc.top);

        memDC.MoveTo(rc.right, nTop);

        memDC.LineTo(rc.right, rc.top);

        memDC.DrawText(csLFinality, &rc, DT\_LEFT| DT\_VCENTER | DT\_SINGLELINE);

        memDC.DrawText(csRFinality, &rc, DT\_RIGHT| DT\_VCENTER | DT\_SINGLELINE);

        memDC.SelectObject(oldfont);

        memDC.SelectObject(hPenOld);

    }

    TitleFont.DeleteObject();

    DetailFont.DeleteObject();

    cPen.DeleteObject();

}