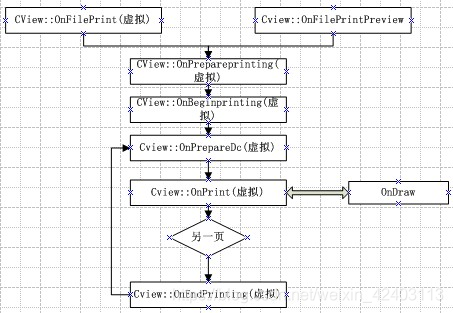
**VS2017 MFC对话框程序打印及打印预览的实现**

花了一个星期，研究了网上大量的MFC对话框打印及打印预览功能的demo之后，网上现有的版本都比较旧，所以选中了几个版本，合并修改，得到这个比较完美的最新版本，编译无错，成功运行。  
MFC文档/视图的应用程序，向导给了打印及打印预览的标准支持。使这类应用程序打印及打印预览工作得以简化。另一类对话框程序却没有相应支持，从MFC打印及打印预览的标准支持入手，可以在对话框程序中，增加三个类以支持打印及打印预览，本文介绍了这三个类的实现。  
打印及打印预览是编写应用程序经常要解决的问题，为了理解VC++对话框程序的打印及打印预览实现,要先掌握基于文档/视图的应用程序打印及打印预览的基本原理。所以分为两部分介绍。  
一、基于文档/视图的应用程序的打印及打印预览原理  
VC++基于文档/视图的应用程序中用MFC应用程序向导在步骤4对话框中选中Print and Print Preview选项，可以包含基本打印及打印预览的支持，应用程序文件菜单中会生成两个菜单项分别是打印（标识符ID\_FILE\_PRINT）和打印预览(标识符：ID\_FILE\_PRINT\_PREVIEW)，展开程序源代码，可以发现，是Cview类提供标准打印和打印预览菜单命令的消息处理函数：  
设应用程序视图类为CMyView,展开MyView.cpp,其消息映象部分有如下两行：

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

ON\_COMMAND(ID\_FILE\_PRINT\_PREVIEW,CView::OnFilePrintPreview)

CView::OnFilePrint 和CView::OnFilePrintPreview函数都进行打印操作，但View::OnFilePrint将实际发送到打印机，而CView::OnFilePrintPreview则将输出发送到程序窗口上方显示的预览窗口显示一个或两个打印页面的复制外观。利用上面加入的缺省打印支持，只能打印或预览图形的一个打印页面，一个页中放不下的部分则放弃，为了加强程序，使它打印整个图形，一页中放不下的部分放在另一页，可以通过覆盖几个打印期间调用的虚拟函数来完成。下图说明了整个打印和打印预览的过程，显示了每个虚拟函数在程序中哪个部分调用。注意每打印一个页面就要经历图中的一个循环。  


二、对话框程序打印及打印预览的实现  
以上是基于文档/视图的应用程序的打印原理，可以根据需要从CView类派生出视图类覆盖打印及打印预览过程中的CView类的虚拟函数来定制文档/视图应用程序的打印及打印预览。在实际中，有很多基于对话框的应用程序，也需要提供打印及打印预览。但向导没有给基于对话框应用程序的基本支持。有了以上知识，可以构造出无文档的视图类，具体的实现时，增加三个类，用以支持打印及打印预览。以下用一具体实例说明。

| **函数名** | **覆盖函数可能完成的任务** |
| --- | --- |
| CView::OnPreparePrinting() | 调用CprintInfo成员函数（如CprintInfo::SetMaxPage设置文档长度）或设置CprintInfo数据成员以影响Print对话框或打印预览操作，然后调用DoPreparePrinting生成用于打印或打印预览的设备描述表（注意必须覆盖OnPreparePrinting并调用DoPreparePrinting） |
| CView::OnBeginPrinting() | 分配专门用于打印的字体，画笔、画刷和其它对象，根据设备描述表计算并设置文档长，在设备描述表上存放所需的消息（这是第一个访问设备描述表的的虚拟函数） |
| CView::OnPrepareDC() | 设置打印当前页面的文本或图形属性，修改视图原点，以打印当前页面，如果没有设置文档长度，在文档末尾终止打印循环（CprintInfo::m\_bContinuePrinting赋值FALSE） |
| CView::OnPrint() | 调用OnDraw进行输出；调用OnDraw前选择OnBeginPrinting分配的字体，调用OnDraw后取消对象，打印只出现在文档打印版中的页头和页脚，如果打印输出与屏幕输出的外观不同， 在这里打印，而不是调用OnDraw |
| CView::OnEndPrinting() | 调用Cgdi::DeleteObject删除OnBeginPrinting分配的对象 |

1.用MFC应用向导创建对话框应用程序，设主对话框类为CPrintPreviewDlg，在主对话框上放一按钮，（标题：打印预览，ID：IDC\_PRINTPREVIEW\_BUTTON），用类向导增加其BN\_CLICKED的消息响应函数OnPrintPreviewButton生成打印预览界面

void CPrintPreviewDlg::OnPrintPreviewButton()

{

CPrintFrame\* pf = new CPrintFrame(this);

}

2、增加新类：  
增加新类步骤：项目->增加新项->MFC->MFC类  
用ClassWizard新建CPrintFrame类（基类CFrameWnd）,功能上相当于文档/视图的应用程序的框架窗口类。  
用ClassWizard新建CPrintView类（基类CScrollView）,功能上相当于文档/视图的应用程序的视图类。  
增加CPrintPreviewView类（基类CPreviewView,注意在头其定义头文件中加入包含afxpriv.h),用于打印预览界面的视图类。（如果没有基类CPreviewView，只有CPreviewViewEx基类，手动将CPreviewViewEx类改成基类CPreviewView）

3、对新生成的各类修改如下：  
CPrintFrame类：

CPrintFrame.h:  
#define WM\_BEGIN\_PRINTING (WM\_USER+1004)

1、增加公有数据成员

CPrintPreviewDlg\* m\_pOldWnd; // 用于保存主对话框对象；

CPrintView\* m\_pView; // 用于保存视图类对象；

2、重载构造函数，保存对主对话框对象指针，创建窗口  
CPrintFrame.h:

CPrintFrame(); // 动态创建所使用的受保护的构造函数

CPrintFrame(CPrintPreviewDlg\* pOld);

virtual ~CPrintFrame();

CPrintFrame.cpp:

CPrintFrame::CPrintFrame(CPrintPreviewDlg\* pOld)

{

m\_pOldWnd = pOld;

if ( !Create(NULL, "打印预览", WS\_OVERLAPPEDWINDOW | FWS\_ADDTOTITLE, CRect(200, 200, 500, 500)))

TRACE0("Failed to create view window! \n");

}

3、修改析构函数，让主对话框显示

CPrintFrame::~CPrintFrame()

{

m\_pOldWnd->ShowWindow(SW\_SHOW);

}

4、添加打印模式函数，设置A4纸纵向打印或横向打印(可以选择使用)

#define DMORIENT\_PORTRAIT 1 //纵向

#define DMORIENT\_LANDSCAPE 2 //横向

void SetLandscapeMode(int PrintMode)

{

PRINTDLG pd;

pd.lStructSize=(DWORD)sizeof(PRINTDLG);

BOOL bRet=AfxGetApp()->GetPrinterDeviceDefaults(&pd);

if(bRet)

{

// protect memory handle with ::GlobalLock and ::GlobalUnlock

DEVMODE FAR \*pDevMode=(DEVMODE FAR \*)::GlobalLock(pd.hDevMode);

pDevMode->dmPaperSize=DMPAPER\_A4; //将打印纸设置为A4

// set orientation to landscape

if(PrintMode==1) //纵向打印

pDevMode->dmOrientation = DMORIENT\_PORTRAIT;

else if(PrintMode==2) //横向打印

pDevMode->dmOrientation = DMORIENT\_LANDSCAPE;

::GlobalUnlock(pd.hDevMode);

}

}

5、用类向导增加WM\_Create消息处理函数（关联CPrintView视图对象；调用其OnFilePrintPreview函数进行打印预览（若要直接打印，可直接向其发送消息）；隐藏主对话框。）

int CPrintFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)

{

if (CFrameWnd::OnCreate(lpCreateStruct) == -1)

return -1;

// TODO: 在此添加您专用的创建代码

CCreateContext context;

context.m\_pNewViewClass = RUNTIME\_CLASS(CPrintView);

context.m\_pCurrentFrame = this;

context.m\_pCurrentDoc = NULL;

context.m\_pLastView = NULL;

m\_pView = STATIC\_DOWNCAST(CPrintView, CreateView(&context));

if (m\_pView != NULL)

{

m\_pView->ShowWindow(SW\_SHOW);

SetActiveView(m\_pView);

// SetLandscapeMode(DMORIENT\_LANDSCAPE);

}

SetIcon(m\_pOldWnd->GetIcon(FALSE), FALSE);

SetIcon(m\_pOldWnd->GetIcon(TRUE), TRUE);

ShowWindow(SW\_RESTORE);

CWinApp \*pApp = AfxGetApp();

pApp->m\_pMainWnd = this;

m\_pView->OnFilePrintPreview();//打印预览

m\_pView->SendMessageW(WM\_COMMAND, ID\_FILE\_PRINT); // 直接打印

m\_pOldWnd->ShowWindow(SW\_SHOW);

return 0;

}

6、添加WM\_Close消息处理函数`

void CPrintFrame::OnClose()

{

// TODO: Add your message handler code here and/or call default

CPrintFrame\* pf=(CPrintFrame\*)::AfxGetMainWnd();

CWinApp \*pApp=AfxGetApp();

pApp->m\_pMainWnd = pf->m\_pOldWnd;

pf->DestroyWindow();

// CFrameWnd::OnClose();

}

7、添加WM\_DESTROY消息处理函数

{

if (m\_pView != NULL)

m\_pView->DestroyWindow();

CFrameWnd::OnDestroy();

CFrameWnd::OnDestroy();

}

CPrintView类：

①修改构造函数：将坐标射模式置为缺省模式。

CPrintView::CPrintView()

{

m\_nMapMode = MM\_TEXT;

}

②增加消息映射实现打印。

BEGIN\_MESSAGE\_MAP(CPrintView, CScrollView)

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

END\_MESSAGE\_MAP()

③重载虚函数OnPreparePrinting，调用DoPreparePrinting生成用于打印或打印预览的设备描述表。

BOOL CPrintView::OnPreparePrinting(CPrintInfo\* pInfo)

{

return DoPreparePrinting(pInfo);

}

④增加公有函数OnFilePrintPreview，调用DoPrintPreview实现打印预览，该函数需要传入四个参数：打印预览工具条资源ID，执行打印及打印预览的视图对象指针，打印预览界面视图类的 CRuntimeClass指针，打印预览状态类CPrintPreviewState对象指针。

void CPrintView::OnFilePrintPreview()

{

// TODO: 在此处添加实现代码.

CPrintPreviewState\* pState = new CPrintPreviewState;

pState->lpfnCloseProc = \_AfxPrintPreviewCloseProc; //设置打印预览窗口关闭时的调用函数

if (!DoPrintPreview(AFX\_IDD\_PREVIEW\_TOOLBAR, this, RUNTIME\_CLASS(CPrintPreviewView), pState))

{

TRACE0("Error, DoPrintPreview failed. \n");

AfxMessageBox(AFX\_IDP\_COMMAND\_FAILURE);

delete pState;

}

}

⑤一些其他函数的实现

void CPrintView::OnInitialUpdate()

{

CScrollView::OnInitialUpdate();

CSize sizeTotal;

// TODO: 计算此视图的合计大小

sizeTotal.cx = sizeTotal.cy = 100;

SetScrollSizes(MM\_TEXT, sizeTotal);

}

void CPrintView::OnDraw(CDC\* pDC)

{

CDocument\* pDoc = GetDocument();

}

void CPrintView::OnBeginPrinting(CDC\* pDC, CPrintInfo\* pInfo)

{

// TODO: 在此添加专用代码和/或调用基类

CView::OnBeginPrinting(pDC, pInfo);

CPrintFrame \*pFrame = (CPrintFrame \*)GetParent();

pFrame->m\_pOldWnd->SendMessage(WM\_BEGIN\_PRINTING, (WPARAM)pDC, (LPARAM)pInfo);

}

void CPrintView::OnEndPrinting(CDC\* pDC, CPrintInfo\* pInfo)

{

// TODO: 在此添加专用代码和/或调用基类

CScrollView::OnEndPrinting(pDC, pInfo);

if (m\_fontPrinter.m\_hObject != NULL)

m\_fontPrinter.DeleteObject();

return;

}

⑥在PrintView.cpp文件中增加全局函数，\_AfxMyPreviewCloseProc，当单击打印预览窗口关闭按钮时被调用。

BOOL CPrintView::\_AfxPrintPreviewCloseProc(CFrameWnd \* pFrameWnd)

{

ASSERT\_VALID(pFrameWnd);

CPrintPreviewView\* pView = (CPrintPreviewView\*)pFrameWnd->GetDlgItem(AFX\_IDW\_PANE\_FIRST);

ASSERT\_KINDOF(CPreviewView, pView);

pView->OnPreviewClose();

return FALSE;

}

7、增加公有函数OnPrint(CDC\* pDC, CPrintInfo\* pInfo)，添加自己需要打印的文字和图片。  
CPrintView.h:

CDC\* m\_dc;

int m\_page\_v\_margin; //纵向边距

int m\_page\_h\_margin; //横向边距

int m\_page\_height; //页高,包括margin

int m\_page\_width; //页宽,包括margin

int m\_current\_y\_pos; //纵向当前打印到哪个位置了，如果到了页底部要调用EndPage重新开一页

int m\_line\_height; //行高，包括字体高度和行间距

int m\_v\_dist; //行间距

BOOL m\_is\_printing\_page;//正在打印页

BOOL m\_mission\_started; //打印任务已经开始

BOOL m\_need\_start\_new\_page;//需要开始新的一页

int m\_max\_page; //总页码

int m\_page\_number; //当前页码

int m\_total\_line\_number; //要打印的内容的总行数，不包括页脚页眉，只包括数据行

int m\_max\_line\_count\_one\_page;

CString m\_doc\_name; //页眉文字

CPrintView.cpp:

void CPrintView::OnPrint(CDC\* pDC, CPrintInfo\* pInfo)

{

// TODO: 在此添加专用代码和/或调用基类

CScrollView::OnPrint(pDC, pInfo);

m\_dc = pDC;

m\_page\_v\_margin = 0;

m\_page\_h\_margin = 0;

m\_page\_height = 0;

m\_current\_y\_pos = 0;

m\_line\_height = 0;

m\_is\_printing\_page = FALSE;

m\_mission\_started = FALSE;

m\_page\_number = 0;

m\_max\_page = 0;

m\_total\_line\_number = 0;

m\_v\_dist = 2;

m\_max\_line\_count\_one\_page = 0;

m\_doc\_name = AfxGetAppName();

m\_need\_start\_new\_page = TRUE;

TEXTMETRIC tm;

m\_dc->GetTextMetrics(&tm);

m\_line\_height = tm.tmHeight + tm.tmInternalLeading;

m\_page\_v\_margin = m\_dc->GetDeviceCaps(LOGPIXELSY) / 2;

m\_page\_h\_margin = m\_dc->GetDeviceCaps(LOGPIXELSX) / 2;

m\_page\_height = m\_dc->GetDeviceCaps(VERTRES);

m\_page\_width = m\_dc->GetDeviceCaps(HORZRES);

m\_current\_y\_pos = m\_page\_v\_margin;

//求出页数

m\_max\_line\_count\_one\_page = (int)((m\_page\_height - m\_page\_v\_margin \* 2 - 50) / (m\_line\_height + 50));

m\_max\_page = (int)(m\_total\_line\_number / m\_max\_line\_count\_one\_page);

if (m\_max\_page\*m\_max\_line\_count\_one\_page < m\_total\_line\_number) m\_max\_page++;

//UINT gl\_uNumOfPoints = 52;

//CDC\* pDC = (CDC\*)wParam;

//CPrintInfo\* pInfo = (CPrintInfo \*)lParam;

int nPageNumber = pInfo->m\_nCurPage;

int i, j;

CFont \*pOldFont;

CFont DataFont;

DataFont.CreatePointFont(120, \_T("宋体"), m\_dc);

pOldFont = m\_dc->SelectObject(&DataFont);

if (1)

{

TCHAR \*pszTitle[200] = { \_T("编号"),\_T("数值1"),\_T("数值2"),\_T("数值3"),\_T("数值4"),\_T("数值5"),\_T("数值6") };

pOldFont = m\_dc->SelectObject(&DataFont);

//画矩形

m\_dc->Rectangle(m\_page\_h\_margin, m\_page\_v\_margin, m\_page\_width - m\_page\_h\_margin, m\_page\_height - m\_page\_v\_margin);//bottom

for (int i = 0; i<7; i++)

{

m\_dc->TextOut(

m\_page\_h\_margin + 50 + i \* (m\_page\_width - m\_page\_h\_margin \* 2) / 7,

4 \* m\_line\_height,

CString(pszTitle[i]));

}

m\_dc->SelectObject(pOldFont);

}

}

CPrintPreviewView类：

手动修改添加类CPrintPreviewView继承CPreviewView类

CPrintPreviewView.h:

#pragma once

#include "afxpriv.h"

#if !defined(AFX\_MYPREVIEWVIEW\_H\_\_0AE8B670\_B1AE\_11DA\_812E\_00E04C39032F\_\_INCLUDED\_)

#define AFX\_MYPREVIEWVIEW\_H\_\_0AE8B670\_B1AE\_11DA\_812E\_00E04C39032F\_\_INCLUDED\_

#if \_MSC\_VER > 1000

#pragma once

#endif // \_MSC\_VER > 1000

#include <afxpriv.h>

// CPrintPreviewView 视图

class CPrintPreviewView : public CPreviewView

{

DECLARE\_DYNCREATE(CPrintPreviewView)

protected:

CPrintPreviewView(); // 动态创建所使用的受保护的构造函数

virtual ~CPrintPreviewView();

public:

#ifdef \_DEBUG

virtual void AssertValid() const;

#ifndef \_WIN32\_WCE

virtual void Dump(CDumpContext& dc) const;

#endif

#endif

public:

afx\_msg void OnPreviewClose();

protected:

afx\_msg void OnPreviewPrint();

DECLARE\_MESSAGE\_MAP()

virtual void OnDraw(CDC\* pDC);

virtual void OnEndPrintPreview(CDC\* pDC, CPrintInfo\* pInfo, POINT point, CPreviewView\* pView);

};

增加工具栏按钮的消息响应函数OnPreviewClose()，OnPreviewPrint() .cpp如下详尽代码

#include "stdafx.h"

#include "PrintPreview.h"

#include "CPrintPreviewView.h"

#include "CPrintFrame.h"

// CPrintPreviewView

IMPLEMENT\_DYNCREATE(CPrintPreviewView, CPreviewView)

CPrintPreviewView::CPrintPreviewView()

{

}

CPrintPreviewView::~CPrintPreviewView()

{

}

BEGIN\_MESSAGE\_MAP(CPrintPreviewView, CPreviewView)

ON\_COMMAND(AFX\_ID\_PREVIEW\_CLOSE, OnPreviewClose)

ON\_COMMAND(AFX\_ID\_PREVIEW\_PRINT, OnPreviewPrint)

END\_MESSAGE\_MAP()

// CPrintPreviewView 诊断

#ifdef \_DEBUG

void CPrintPreviewView::AssertValid() const

{

CPreviewView::AssertValid();

}

#ifndef \_WIN32\_WCE

void CPrintPreviewView::Dump(CDumpContext& dc) const

{

CPreviewView::Dump(dc);

}

void CPrintPreviewView::OnPreviewClose()

{

CPrintFrame\* pf = (CPrintFrame\*)::AfxGetMainWnd();

CWinApp \*pApp = AfxGetApp();

pApp->m\_pMainWnd = pf->m\_pOldWnd;

pf->DestroyWindow();

}

void CPrintPreviewView::OnPreviewPrint()

{

m\_pPrintView->SendMessage(WM\_COMMAND, ID\_FILE\_PRINT);

}

#endif

#endif //\_DEBUG

// CPrintPreviewView 消息处理程序

void CPrintPreviewView::OnDraw(CDC\* pDC)

{

// TODO: 在此添加专用代码和/或调用基类

CPreviewView::OnDraw(pDC);

m\_pToolBar->PostMessage(WM\_IDLEUPDATECMDUI, (WPARAM)TRUE);// 控制条的命令状态更新

}

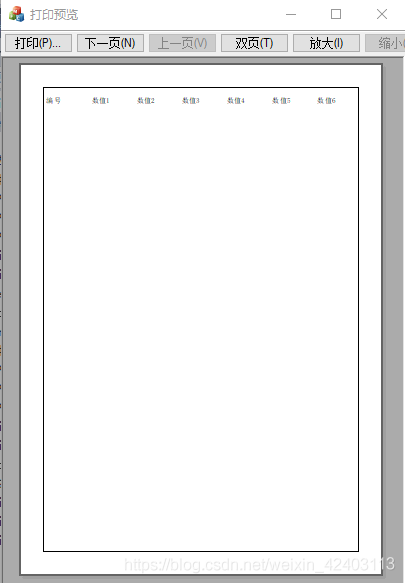
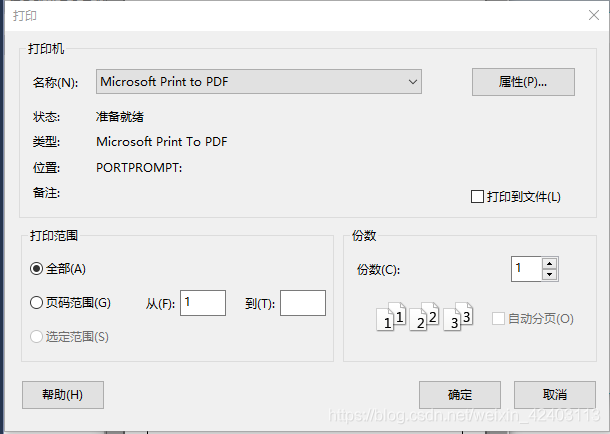
void CPrintPreviewView::OnEndPrintPreview(CDC\* pDC, CPrintInfo\* pInfo, POINT point, CPreviewView\* pView)

{

// TODO: 在此添加专用代码和/或调用基类

CPreviewView::OnEndPrintPreview(pDC, pInfo, point, pView);

}

最终结果图;  
  
  
  
程序链接：<https://download.csdn.net/download/weixin_42403113/11069726>