在Windows应用程序开发过程中，实现文档打印功能是每一个开发人员都不得不面对的问题。在使用Visual C++6.0这一开发工具的情况下，虽然它的AppWizard生成的应用程序框架中缺省地包括打印功能，但是如何具体实现打印功能的确是一件非常繁杂的事情，尤其是要实现文档的分页打印，更是一个让很多初学者感到困惑的问题。笔者结合自己学习体会，谈一谈如何实现文档的分页打印功能，希望可以对广大读者朋友们起到一定的帮助。  
  
　　**一、MFC程序默认的打印流程**  
　　Visual C++6.0的应用程序向导（AppWizard）生成的应用程序具备了实现打印功能的基本框架，当用户从菜单中选择Print命令时，应用程序就用ID\_FILE\_PRINT命令消息调用程序视图类的缺省处理函数OnFilePrint（）来开始打印过程，并创建用语打印的设备环境，来满足应用程序特定的需要。其中要重载应用程序视图类的几个相应函数，其函数表如图一所示：

|  |  |
| --- | --- |
| 函数 | 说明 |
| OnBeginPrinting（） | 重载此函数来创建在打印文档时所需的资源，如字体等。在此也可以设置最大页数。 |
| OnDraw（） | 该函数提供三重服务，在框架窗口、预览窗口和打印机上显示数据，具体在哪里显示，依赖于作为函数参数传递的设备环境。 |
| OnEndPrinting（） | 重载此函数用来释放在OnBeginPrinting（）中创建的资源。 |
| OnPrepareDC（） | 重载此函数来修改用于显示或打印文档的设备环境，可以在此处理换页。 |
| OnPreparePrinting（） | 提供打印文档的最大页数，如果在此不设置页数，必须在OnBeiginPrinting（）中设置。 |
| OnPrint（） | 提供附加打印服务，如在Cview::OnDraw（）函数中不提供的打印页眉、页脚。 |

　　　　　　　　　　　　　　　表一、实现打印功能需要重载的几个CView类成员函数  
  
　　OnFilePrint（）函数一开始就调用OnPreparePrinting（）函数，该函数缺省时则调用DoPreparePrinting（）函数来显示打印对话框，创建的OnPreparePrinting（）程序如下：

|  |
| --- |
| BOOL CmyView::OnpreparePrinting(CprintInfo \*pInfo) { 　//default preparetion 　return DoPreparePrinting(pInfo); } |

　　程序中，OnPreparePrinting（）提供一个指向CprintInfo对象的指针参数，使用此对象，应用程序能够获取关于打印作业的信息，同时对属性（例如最大页数）进行初始化。DoPreparePrinting（）显示对话框时，由用户设置CprintInfo类的许多数据成员变量供应用程序使用或再设置，即用于创建整个打印的设备环境。  
  
　　打印过程都要用到CprintInfo结构，它里面保存了打印任务的信息。OnPreparePrinting（）重载是为了指明打印文件的长度。在调用Cview：：DoPreparePrinting（）之前，可用其参数来修改CprintInfo结构。重载OnPreparePrinting（）还可以使用CprintInfo结构的成员函数SetMaxPage（）、SetMinPage（）来设置文档的最大、最小页数。当把CprintInfo结构中表明文档是否被直接打印标志变量m\_bDirect设置为TRUE时，调用DoPreparePrinting（）就不再显示打印对话框。此时，由于没有打印对话框，可以用SetToPage（）和SetFromPage（）设置打印的起始页。  
  
　　接下来，打印程序调用视图类的OnBegingPrinting（）函数，该函数进一步进行初始化工作，并传送一个指向设备环境的指针和一个指向CprintInfo结构的指针。  
  
　　在开始逐页打印之前，调用OnPrepareDC（）函数，这是每一页执行一次的打印循环的开始，该函数的修改用于显示或打印文档的设备环境，为每页设置映像模式，创建字体，检查文档是否结束。可以把CPrintInfo结构中的m\_bContinuePrinting标志设置为False来发出打印完毕信号以终止打印作业。  
  
　　在调用了OnPrepareDC（）函数之后，ID\_FILE\_PRINT消息处理函数调用了OnPrint（）函数，该函数的缺省功能是把打印工作放在视图类的OnDraw（）函数中进行，它传送一个指向设备环境的指针和一个指向CprintInfo结构的指针，负责每一页的打印。  
  
　　归纳起来，MFC应用程序的打印流程如下：

|  |
| --- |
| 图一、打印流程图 |

　　**二、文档分页打印功能的实现**  
  
　　为了说明如何实现文档的分页打印功能，下面给出了分页打印输出几个矩形的实例。首先启动Visual C++6.0生成一个应用程序框架，并重载视图类的OnBeginPrinting（）函数，在该函数中获取设备环境信息，设置页高、页数，由此设置打印对话框的初值，具体代码如下：

|  |
| --- |
| Void CmyView::OnBeginPrinting(CDC\*pdc,CprintInfo \*pInfo) { 　CmyDoc \*pDoc=GetDocment（）；//得到文档指针； 　ASSERT\_VALID（pDoc）；//确认指针的有效性； 　int pageHeitht=pDC->GetDeviceCaps(VERTRES)；//由页顶到页底可打印点的个数，从设备环境中得到页面高度； 　int logPixelsy=pDC->GetDeviceCaps(LOGPEXELSY)；//每英寸的点数； 　int rectHeight=(int)(4.2\*logPixelsy)；//计算矩形的高度，可以由用户自行设置； 　int numPages=Pdoc->m\_numRects\*rectHeight/pageHeight+1；//由矩形数计算输出页数； 　pInfo->SetMaxPage(numPages)；//初始化pInfo结构； } |

　　为了将矩形打印输出，还应该重载Cview：：OnPrepareDC（）函数，实现输出图形的位置定位，函数首先测试应用程序是否正在调用CDC类的成员函数IsPrinting（）打印数据，若正在打印，则判断当前页的数据属于哪一部分，由GetDeviceCaps（）函数得当环境设备信息；接着确定打印新视图的原点，并将原点值由SetViewportOrg（）传递给设备环境。具体代码如下：

|  |
| --- |
| Void CmyView::OnPrepareDC(CDC\*pDC，CprintInfo \*pInfo) { 　if(pDC->IsPrinting())//若正在进行打印则判断当前页数据； 　{ 　　int pageHeight=pDC->GetDeviceCaps(VERTRES); 　　int originY=pageHeight\*((pInfo->m\_nCurpage-1); 　　pDC->SetViewportOrg(0,-originY) 　} 　Cview::OnPrepareDC(pDC,pInfo); } |

　　**三、小结**  
  
　　通过上文的介绍，相信读者朋友应该对Visual C++编程实现文档的分页打印功能有了一定的了解，大家可以在此基础上，参考相关学习资料，根据应用程序的不同需要，编制实现不同的打印程序。