Overview

Introduction

PrintForm is a module for printing and previewing structured documents.
Programmers can use it to generate nice-looking multipage documents with little effort.
It is very well suited for database reports and spreadsheets.

PrintForm documents consist of a header-, footer- and body area. Within each area you can have PrintItems. Each of them can contain one bitmap, icon or string.
The strings:
- can force a form-feed,
- can have their own font,
- can be word-wrapped in their area,
- can be aligned left, centered and right and,
- within headers or footers, they can contain page numbers.

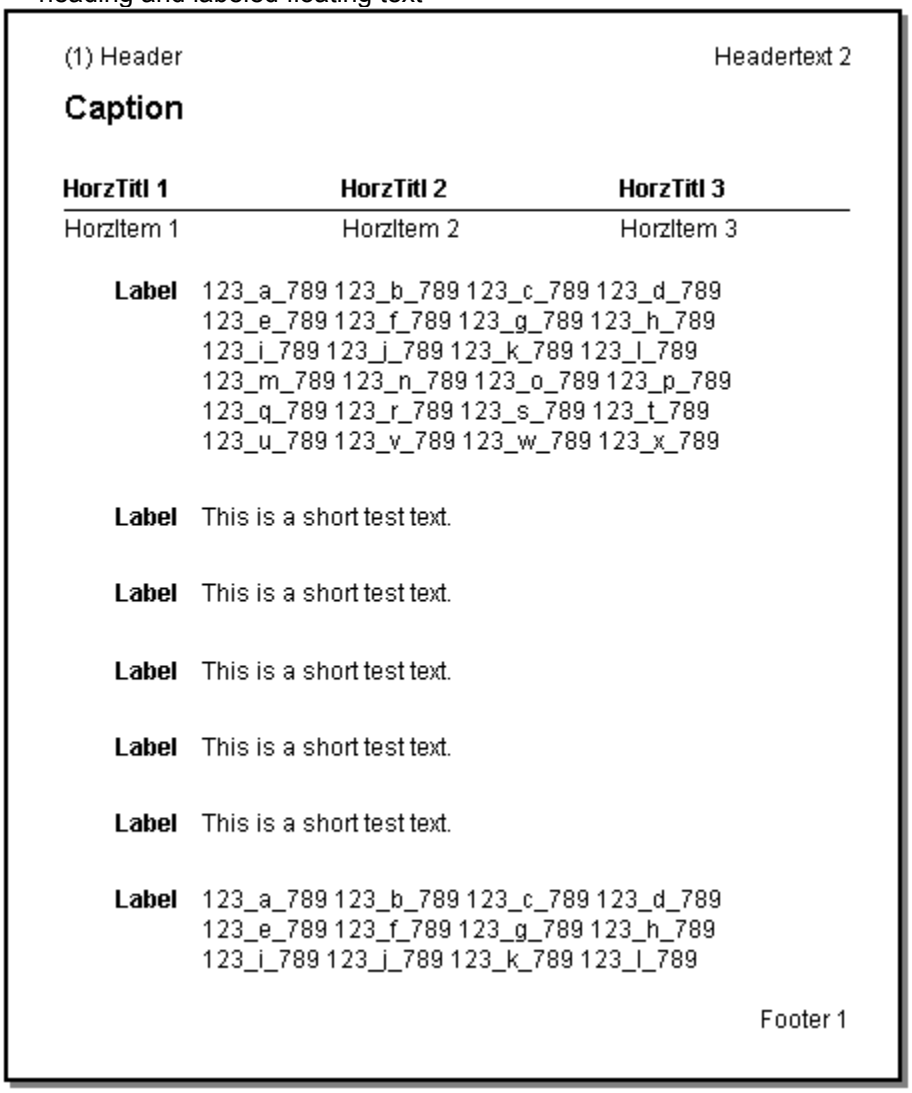
The positioning of the PrintItems is very dynamic. Each PrintItem has a vertical offset, a horizontal position and a fixed width. The height can be computed from the height of the word-wrapped text.

Examples

- Report Example
- Spreadsheet Example

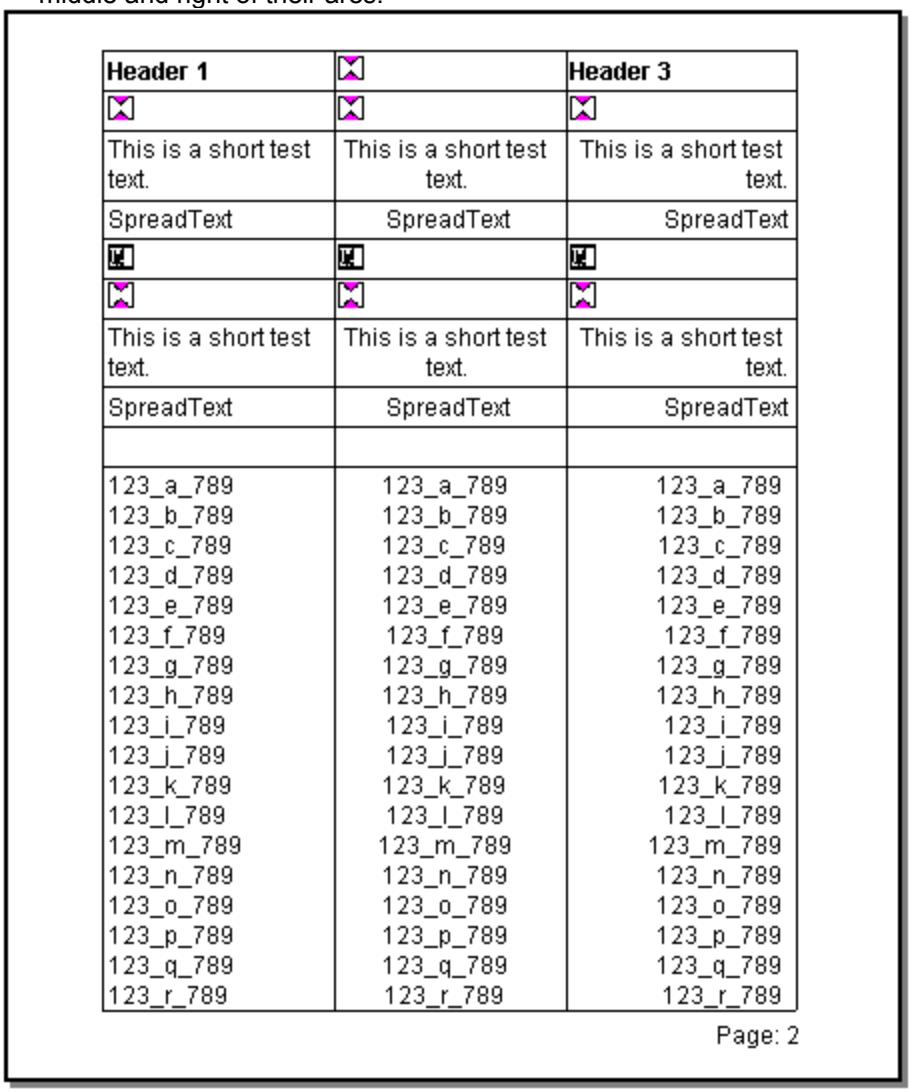
Report Example

The following example contains simple headers and footers and a body are with a multi-column heading and labeled floating text



Spreadsheet Example

The following document has borders and bitmaps. The header contains matching column headers for the body text. There are short and very long strings and the are aligned to the left, middle and right of their ares.



Header 1		Header 3
This is a short test text.	This is a short test text.	This is a short test text.
SpreadText	SpreadText	SpreadText
This is a short test text.	This is a short test text.	This is a short test text.
SpreadText	SpreadText	SpreadText
123_a_789	123_a_789	123_a_789
123_b_789	123_b_789	123_b_789
123_c_789	123_c_789	123_c_789
123_d_789	123_d_789	123_d_789
123_e_789	123_e_789	123_e_789
123_f_789	123_f_789	123_f_789
123_g_789	123_g_789	123_g_789
123_h_789	123_h_789	123_h_789
123_i_789	123_i_789	123_i_789
123_j_789	123_j_789	123_j_789
123_k_789	123_k_789	123_k_789
123_l_789	123_l_789	123_l_789
123_m_789	123_m_789	123_m_789
123_n_789	123_n_789	123_n_789
123_o_789	123_o_789	123_o_789
123_p_789	123_p_789	123_p_789
123_q_789	123_q_789	123_q_789
123_r_789	123_r_789	123_r_789

Page: 2

Programming

- Abstract
- Reference
- General
- Functions
- PfoInit
- PfoPreparePrinting
- PfoBeginPrinting
- PfoPrepareDC
- PfoPrint
- PfoEndPrinting
- PfoTrace
- PfoExit
- Data Structures
- CFontHandleArray CFontHandleArray
- CPrintItemArray CPrintItemArray
- CPrintItem

Abstract

You start by defining the normal MFC printing functions in you code. They are:

OnPreparePrinting
OnBeginPrinting
OnPrepareDC
OnPrint
OnEndPrinting

You might want to use the application wizard to do that. See the following chapter for complete code examples.

Within these functions you will call one or two PrintForm functions. Most of them have names similar to the MFC functions. They are:

PfoInit
PfoPreparePrinting
PfoBeginPrinting
PfoPrepareDC
PfoPrint
PfoEndPrinting
PfoTrace
PfoExit

Most of these function need a pPrintForm as parameter. This is the PrintForm object and it is allocated by PfoInit. For PfoInit needs the following data to do its work:
- an array of font handles,
- the page rect and the relative heights of the header and footer area.
- one array of print items for the header, footer and body text.

Thats all you have to do to create PrintForm documents.

Reference

- General
- Functions
- PfoInit
- PfoPreparePrinting
- PfoBeginPrinting
- PfoPrepareDC
- PfoPrint
- PfoEndPrinting
- PfoTrace
- PfoExit
- Data Structures
- CFontHandleArray CFontHandleArray
- CPrintItemArray CPrintItemArray
- CPrintItem

General

All measurements are in logical GDI coordinates. You have to specify a mapping mode.

Functions

- PfoInit
- PfoPreparePrinting
- PfoBeginPrinting
- PfoPrepareDC
- PfoPrint
- PfoEndPrinting
- PfoTrace
- PfoExit

PfoInit

```
UINT EXPENTRY PfoInit(
    CPrintForm**            ppPrintForm,
    const CWordArray*       pcaFontHandles,
    const CRect*            pcrcPage,
    const UINT              cuiHeaderDY,
    const UINT              cuiFooterDY,
    const CPrintItemArray*  pcapriHeader,
    const CPrintItemArray*  pcapriFooter,
    const CPrintItemArray*  pcapriBody );
```

The PfoInit function generates the PrintForm object and initializes it. It is used within OnPreparePrinting.

Parameter	Description
ppPrintForm	The address of the PrintForm pointer. The PrintForm object is allocated and the pointer to it is stored here.
pcaFontHandles	A pointer to a constant array of font handles. The application creates the font handles and stores them in a simple CWordArray.
pcrcPage	A pointer to a constant rect that contains the dimensions of the page. All other position and offset values are relative to the margins that are specified by this rect. The page area (without the margins) starts at the logical (0,0) GDI position, that is, not at the physical top-left paper edge.
cuiHeaderDY	A constant UINT with the height of the header. The header area is within the margins that are specified by pcrcPage.
cuiFooterDY	A constant UINT with the height of the footer. The footer area is within the margins that are specified by pcrcPage.
pcapriHeader	A constant array of PrintItems for the header. The CPrintItemArray is described below.
pcapriFooter	A constant array of PrintItems for the footer. The CPrintItemArray is described below.
pcapriBody	A constant array of PrintItems for the body text. The CPrintItemArray is described below.

Example

```
BOOL CPfoView::OnPreparePrinting(CPrintInfo* pInfo)
{
    if( DoPreparePrinting( pInfo ) ) {

       CPfoDoc* pDoc = GetDocument();

       if( 0 == PfoInit( & m_pPrintForm,
                    & pDoc->m_awFontHandles,
                    & pDoc->m_rcPage,
                      pDoc->m_uiHeaderDY,
                      pDoc->m_uiFooterDY,
                    & pDoc->m_apriHeader,
                    & pDoc->m_apriFooter,
                    & pDoc->m_apriBody ) )
        {
            if( 0 == PfoPreparePrinting( m_pPrintForm, pInfo ) ) {
                return TRUE;
            }
        }
    }
    return FALSE;
}
```

PfoPreparePrinting

```
UINT EXPENTRY PfoPreparePrinting(
    CPrintForm*             pPrintForm,
    CPrintInfo*             pInfo );
```

The PfoPreparePrinting function is used in OnPreparePrinting after PfoInit has been called.

Parameter	Description
pPrintForm	The pointer to the PrintForm object.
pInfo	The pointer to the CPrintInfo structure of MFC.

Example

```
See PfoInit
```

PfoBeginPrinting

```
UINT EXPENTRY PfoBeginPrinting(
    CPrintForm*             pPrintForm,
    CDC*                    pDC,
    CPrintInfo*             pInfo );
```

This functions is called in OnBeginPrinting.

Parameter	Description
pPrintForm	The pointer to the PrintForm object.
pDC	The pointer to the printer or print preview DC.
pInfo	The pointer to the CPrintInfo structure of MFC.

Example

```
void CPfoView::OnBeginPrinting(CDC* pDC, CPrintInfo* pInfo)
{
    PfoBeginPrinting( m_pPrintForm, pDC, pInfo );
}
```

PfoPrepareDC

```
UINT EXPENTRY PfoPrepareDC(
    CPrintForm*             pPrintForm,
    CDC*                    pDC,
    CPrintInfo*             pInfo,
    BOOL*                   pbContinuePrinting );
```

This function is called in OnPrepareDC, but only, if the OnPrepareDC function is called for printing and print preview. The pInfo parameter is not NULL, then.

Parameter	Description
pPrintForm	The pointer to the PrintForm object.
pDC	The pointer to the printer or print preview DC.
pInfo	The pointer to the CPrintInfo structure of MFC.
pbContinuePrinting	A pointer to a boolean flag. This flag is set to TRUE, if there are more pages to print, FALSE otherwise.

Example

```
void CPfoView::OnPrepareDC( CDC* pDC, CPrintInfo* pInfo )
{
    CPfoDoc* pDoc = GetDocument();
    if( pInfo ) {
        // Attention: LOMETRIC means that Y goes upward!!!
        SetScrollSizes( MM_LOMETRIC, pDoc->m_rcPage.Size() );
    } else {
        SetScrollSizes( MM_TEXT, CSize( 0, 0 ) );
    }
    CScrollView::OnPrepareDC( pDC, pInfo );
    if( pInfo ) {
        // Prepare DC and test if there is a page to print.
        PfoPrepareDC( m_pPrintForm, pDC, pInfo, & pInfo->m_bContinuePrinting );
    }
}
```

PfoPrint

```
UINT EXPENTRY PfoPrint(
    CPrintForm*             pPrintForm,
    CDC*                    pDC,
    CPrintInfo*             pInfo,
    CRect                   rcPhysPage );
```

This function is OnPrint. It is, as you might assume, central to the printing process. It needs some preparations, as you can see in the example, below.

Parameter	Description
pPrintForm	The pointer to the PrintForm object.
pDC	The pointer to the printer or print preview DC.
pInfo	The pointer to the CPrintInfo structure of MFC.
rcPhysPage	The physical page rect, in device coordinates. For a printer, this is just the full size of the paper. For preview, this is the area that the simulated page occupies on the physical display screen. See the example for a way to determine this parameter.

Example

```
void CPfoView::OnPrint( CDC* pDC, CPrintInfo* pInfo )
{
    CScrollView::OnPrint( pDC, pInfo );

    CRect rcPhysPage( 0, 0, INT_MAX, INT_MAX );
    if( pInfo->m_bPreview ) {

        // Get the parent window
        CFrameWnd* pMainFrame = (CFrameWnd*) AfxGetApp()->m_pMainWnd;
        ASSERT(pMainFrame != NULL);
        ASSERT(pMainFrame->IsKindOf(RUNTIME_CLASS(CFrameWnd)));
        ASSERT_VALID(pMainFrame);

        // Get the PreviewView
        CMyPreviewView* pPreviewView = (CMyPreviewView*) pMainFrame->GetActiveView();
        if( ISKINDOF( pPreviewView, CPreviewView ) ) {
            // Get the rect of the virtual paper on the screen
            CRect rcScreen = pPreviewView->GetScreenRect( pInfo->m_nCurPage );
            rcPhysPage = rcScreen;
        }
    } else {
        // Get the full printer page
        CPoint pt(  pDC->GetDeviceCaps( HORZRES ),
                    pDC->GetDeviceCaps( VERTRES ) );
        rcPhysPage.SetRect( 0, 0, pt.x, pt.y );
    }

    PfoPrint( m_pPrintForm, pDC, pInfo, rcPhysPage );
}
```

PfoEndPrinting

```
UINT EXPENTRY PfoEndPrinting(
    CPrintForm*             pPrintForm,
    CDC*                    pDC,
    CPrintInfo*             pInfo );
```

This function is called in OnEndPrinting.

Parameter	Description
pPrintForm	The pointer to the PrintForm object.
pDC	The pointer to the printer or print preview DC.
pInfo	The pointer to the CPrintInfo structure of MFC.

Example

```
void CPfoView::OnEndPrinting(CDC* pDC, CPrintInfo* pInfo)
{
    PfoEndPrinting( m_pPrintForm, pDC, pInfo );

    if( m_bDbgTrace ) {
        PfoTrace( m_pPrintForm );
    }
    PfoExit( & m_pPrintForm );

    SetScrollSizes( MM_TEXT, CSize( 0, 0 ) );
}
```

PfoTrace

```
UINT EXPENTRY PfoTrace(
    CPrintForm*             pPrintForm );
```

This is an internal function. See PfoEndPrinting for an example.

PfoExit

```
UINT EXPENTRY PfoExit(
    CPrintForm**            ppPrintForm );
```

This function is called in OnEndPrinting.

Parameter	Description
ppPrintForm	The address of the PrintForm pointer. The PrintForm object is deleted and the pointer to it is cleared.

Example

See PfoEndPrinting

Data Structures

- CFontHandleArray CFontHandleArray
- CPrintItemArray CPrintItemArray
- CPrintItem

CFontHandleArray

```
const CWordArray cFontHandleArray;
```

This is a simple CWordArray that contains the handles of all fonts, that are to be used in the document. The PrintItems (see below) contain indexes into this array and use them to select their fonts.

Example

```
// Create the fonts
STRING_CONSTANT( cszArial, "Arial" );
LOGFONT logfont;
logfont.lfHeight                = 60; // mm/10
logfont.lfWidth                 = 0;
logfont.lfEscapement            = 0;
logfont.lfOrientation           = 0;
logfont.lfWeight                = FW_NORMAL;
logfont.lfItalic                = 0;
logfont.lfUnderline             = 0;
logfont.lfStrikeOut             = 0;
logfont.lfCharSet               = ANSI_CHARSET;
logfont.lfOutPrecision          = OUT_CHARACTER_PRECIS;
logfont.lfClipPrecision         = CLIP_CHARACTER_PRECIS;
logfont.lfQuality               = PROOF_QUALITY;
logfont.lfPitchAndFamily        = VARIABLE_PITCH;
strcpy( logfont.lfFaceName, cszArial );
// Font 1: normal
m_awFontHandles.Add( (WORD) CreateFontIndirect( &logfont ) );
// Font 2: bold, for sub-headings
logfont.lfWeight                = FW_BOLD;
m_awFontHandles.Add( (WORD) CreateFontIndirect( &logfont ) );
// Font 3: larger, bold, for top heading
logfont.lfHeight                = 80;
m_awFontHandles.Add( (WORD) CreateFontIndirect( &logfont ) );
```

CPrintItemArray

```
CPrintItemArray m_apriHeader;
```

The CPrintItemArray is a special case of an CObArray. It contains only pointers to CPrintItems. You add PrintItems to it with the Add(...) method. You can use DeleteAll() to delete all items.

Example

See PrintItem

CPrintItem

This class is central to the formatting process.
It contains most of the data that is used to format the document.
You can create PrintItems with a default or a parametrized constructor.
You could access the member variables directly, by doing so is not recommended.
Here is the parametrized constructor:

```
CPrintItem(
    UINT        uiYOffs,
    UINT        uiXPos,
    UINT        uiWidth,
    UINT        uiHeight,
    UINT        uiType,
    LPCSTR      lpcstr,
    UINT        uiFontIdx,
    UINT        uiAlignement,
    BOOL        bKeepYPos,
    BOOL        bBorderLeft,
    BOOL        bBorderRight,
    BOOL        bBorderTop,
    BOOL        bBorderBottom,
    BOOL        bNewPage );
```

Parameter	Description
uiYOffs	For the first item in an array, this is the vertical offset to the top of the current area. For all other items, the interpretation of this value depends on bKeepYPos: If that flag is on, uiYOffs is the offset from the top of the previous item, otherwise it is the offset from the bottom of it.
uiXPos	The horizontal position of this item, relative to the page rect that is given to PfoInit, but not relative to another print item. This means, that print items might overlap, intentionally, or not.
uiWidth	The width of this print item. This value must always be specified. Item areas may not overlap the right margin of the page rect.
uiHeight	If this value is zero, the height of the item area is determined automatically. For all picture types, it is the height of a single line, for strings it is the height of the text.
uiType	You can use CPrintItem::String, CPrintItem::Bitmap and CPrintItem::Icon.
lpcstr	Depending on uiType, this field contains either a pointer to a C string or the handle of another object in the low word and zero in the high word.
uiFontIdx	An index into the font array. The font array is given as a parameter to PfoInit.
uiAlignement	Here you specify Windows DrawText() flags. The possible values are: DT_LEFT, DT_CENTER, DT_RIGHT, DT_SINGLELINE. You have to specify uiHeight, if you want to use DT_VCENTER or DT_BOTTOM. You don't need to specify: DT_EXPANDTABS, DT_NOPREFIX, DT_WORDBREAK. They are always automatically used.
bKeepYPos	This flag determines the interpretation of uiYOffs, see above.
bBorderLeft	A left border should be drawn. If this flag is on, the width of the item is internally reduced by PFO_BORDER_WIDTH.
bBorderRight	A right border should be drawn. If this flag is on, the width of the item is internally reduced by PFO_BORDER_WIDTH.
bBorderTop	A top border should be drawn. If this flag is on, the height of the item grows by PFO_BORDER_WIDTH.
bBorderBottom	A bottom border should be drawn. If this flag is on, the height of the item grows by PFO_BORDER_WIDTH.
bNewPage	An item with this flag set will appear on a new page.

Contents

- Overview
- Examples
- Report Example
- Spreadsheet Example
- Programming
- Abstract
- Reference
- General
- Functions
- PfoInit
- PfoPreparePrinting
- PfoBeginPrinting
- PfoPrepareDC
- PfoPrint
- PfoEndPrinting
- PfoTrace
- PfoExit
- Data Structures
- CFontHandleArray CFontHandleArray
- CPrintItemArray CPrintItemArray
- CPrintItem