Private Declare Function StretchBlt Lib "gdi32" (ByVal hdc As Long, ByVal X As Long, \_

ByVal Y As Long, ByVal mDestWidth As Long, ByVal mDestHeight As Long, \_

ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc As Long, ByVal mSrcWidth As Long, \_

ByVal mSrcHeight As Long, ByVal dwRop As Long) As Long

Private Const SRCCOPY = &HCC0020

Private Declare Function GetDeviceCaps Lib "gdi32" (ByVal hdc As Long, ByVal nIndex As Long) As Long

'-------------------------------------------------------------------------------------------------------------------

' By using the following messages in VB, it is possible to make a RichTextBox support WYSIWYG display and output:

' EM\_SETTARGETDEVICE message is used to tell a RichTextBox to base its display on a target device.

' EM\_FORMATRANGE message sends a page at a time to an output device using the specified coordinates.

Private Type Rect

Left As Long

Top As Long

Right As Long

Bottom As Long

End Type

Private Type CharRange

firstChar As Long ' First character of range (0 for start of doc)

lastChar As Long ' Last character of range (-1 for end of doc)

End Type

Private Type FormatRange

hdc As Long ' Actual DC to draw on

hdcTarget As Long ' Target DC for determining text formatting

rectRegion As Rect ' Region of the DC to draw to (in twips)

rectPage As Rect ' Page size of the entire DC (in twips)

mCharRange As CharRange ' Range of text to draw (see above user type)

End Type

Private Const WM\_USER As Long = &H400

Private Const EM\_FORMATRANGE As Long = WM\_USER + 57

Private Const EM\_SETTARGETDEVICE As Long = WM\_USER + 72

Const PHYSICALOFFSETX As Long = 112

Const PHYSICALOFFSETY As Long = 113

Private Declare Function SendMessage Lib "user32" Alias "SendMessageA" \_

(ByVal hwnd As Long, ByVal msg As Long, ByVal wp As Long, Ip As Any) As Long

Dim mFormatRange As FormatRange

Dim rectDrawTo As Rect

Dim rectPage As Rect

Dim TextLength As Long

Dim newStartPos As Long

Dim dumpaway As Long

Private Declare Function CreateDC Lib "gdi32" Alias "CreateDCA" \_

(ByVal lpDriverName As String, ByVal lpDeviceName As String, \_

ByVal lpOutput As Long, ByVal lpInitData As Long) As Long

'-------------------------------------------------------------------------------------------------------------------

Dim mNotShow As Boolean

Dim mSizeNo As Integer

Dim mTotalPages As Integer

Private Sub Form\_Load()

Screen.MousePointer = vbHourglass

gprint = False

' we don't want the sizes to change after they have been appropriately sized

PicZ.AutoSize = False ' Base, always visible

picP.AutoSize = False ' For print intermediary, always invisible

PicX.AutoSize = False ' For diaplay intermediary, always invisible

Pic3.AutoSize = False ' As Zoom in

Pic5.AutoSize = False ' As Zoom out

' By default VB prints in twips. If a Picturebox is using pixels, we have to

' convert twips to pixels. Therefore we fix the size of Pictureboxes before

' setting its ScaleMode to pixel (Eash pixel is about 15 twips, depending on

' the resolution of device)

Dim mNormalWidth, mNormalHeight

Dim mAdjFactor

Dim mRect, mNewRect, mfactor

Dim mpage As Integer

' Render document size in line with that of the printer (but note that doc is

' shown on screen without print margins)

DocWYSIWYG frmmain1.RTBox

' Obtain size of the printer

mNormalWidth = Printer.ScaleWidth

mNormalHeight = Printer.ScaleHeight

' Due to diff of resolution between screen and printer, we may use an adjustment

' factor, here we don't have any adjustment

mAdjFactor = 100 / 100

mNormalWidth = mNormalWidth \* mAdjFactor

mNormalHeight = mNormalHeight \* mAdjFactor

' Mark down rectangle area, see remarks later

mRect = mNormalWidth \* mNormalHeight

' Make the invisible PicX of the same size as printer

PicX.Width = mNormalWidth

PicX.Height = mNormalHeight

' Set 75%

Pic3.Width = CInt(mNormalWidth \* 75 / 100)

Pic3.Height = CInt(mNormalHeight \* 75 / 100)

' Set 25%

Pic5.Width = CInt(mNormalWidth \* 25 / 100)

Pic5.Height = CInt(mNormalHeight \* 25 / 100)

' Set ScaleMode to pixels.

preview.ScaleMode = vbPixels

PicZ.ScaleMode = vbPixels

PicX.ScaleMode = vbPixels

Pic3.ScaleMode = vbPixels

Pic5.ScaleMode = vbPixels

' Set AutoRedraw to True

PicZ.AutoRedraw = True

picP.AutoRedraw = True

PicX.AutoRedraw = True

Pic3.AutoRedraw = True

Pic5.AutoRedraw = True

' Set BorderStyle to Fixed Single

PicZ.BorderStyle = 1

PicX.BorderStyle = 1

Pic3.BorderStyle = 1

Pic5.BorderStyle = 1

' Set Fillstyle to Transparent

PicZ.FillStyle = 1

picP.FillStyle = 1

PicX.FillStyle = 1

Pic3.FillStyle = 1

Pic5.FillStyle = 1

' Backcolor of PicZ is blue (&H8000000D), the rest are white (&H80000009)

picP.BackColor = &H80000009

PicX.BackColor = &H80000009

Pic3.BackColor = &H80000009

Pic5.BackColor = &H80000009

' Before showing first page, test how many pages are there in total in RTB.

mTotalPages = PageCtnProc(preview.PicX)

' Display the No. of total pages available

txtTotalPages.Text = "Total " & CStr(mTotalPages) & " pages"

' Enable/disable page movement buttons

setPageButtons

Dim i As Integer

cboPageNo.Clear

For i = 1 To mTotalPages

cboPageNo.AddItem i

Next i

cboPageNo.Text = cboPageNo.List(0)

' Set max of scroll bars

' VScroll1.Max = 1000

' HScroll1.Max = 1000

cboScale.AddItem "Zoom in"

cboScale.AddItem "Zoom out"

cboScale.Text = cboScale.List(1) ' i.e. 25%

' Instead Selprint whole document content such as:

' frmFrame.ActiveForm.ActiveControl.SelPrint preview.picX.Hdc

' we only print a single page at a time. Initially we show page 1.

'

' Whatever page, we will print it to PicX first (then project to other

' pictureboxes according to the sizes they play)

mpage = 1

FormPreviewPage preview.PicX, mpage

' Now stretchblt to wanted sizes.

For i = 1 To 5

DoEvents

If MakeSizes(i) = False Then

Screen.MousePointer = vbDefault

Exit Sub

End If

Next

Screen.MousePointer = vbDefault

' Start display of preview screen.

' Note picZ is always visible, picX always not.

PicZ.Visible = True

picP.Visible = False

PicX.Visible = False

mNotShow = False ' Show appropriate picture on screen

mSizeNo = 5 ' i.e. cboScale.List=4, 25%

ChangePreview

End Sub

Private Sub cboPageNo\_click()

Dim mpage As Integer

mpage = cboPageNo.ListIndex + 1

setPageButtons

Screen.MousePointer = vbHourglass

' Print a new page to PicX

FormPreviewPage preview.PicX, mpage

' Again have to stretchblt to various sizes.

Dim i

For i = 1 To 5

DoEvents

If MakeSizes(i) = False Then

Screen.MousePointer = vbDefault

Exit Sub

End If

Next

' Have to change size (and then change back) to refresh display of new screen

' During the change, not to show any picture, hence mNotShow is temporarily

' set to True

If mSizeNo = 1 Then

mSizeNo = 2

mNotShow = True

ChangePreview

mNotShow = False

mSizeNo = 1

ChangePreview

Else

mSizeNo = mSizeNo - 1

mNotShow = True

ChangePreview

mNotShow = False

mSizeNo = mSizeNo + 1

ChangePreview

End If

Screen.MousePointer = vbDefault

End Sub

Private Sub cmdPrevPage\_Click()

If mTotalPages = 1 Then

Exit Sub

Else

If Val(cboPageNo.Text) > 1 Then

cboPageNo.Text = cboPageNo.List(cboPageNo.ListIndex - 1)

cboPageNo\_click

End If

End If

End Sub

Private Sub cmdNextPage\_Click()

If mTotalPages = 1 Then

Exit Sub

Else

If Val(cboPageNo.Text) < mTotalPages Then

cboPageNo.Text = cboPageNo.List(cboPageNo.ListIndex + 1)

cboPageNo\_click

End If

End If

End Sub

Private Sub setPageButtons()

If mTotalPages = 1 Then

cmdPrevPage.Enabled = False

cmdNextPage.Enabled = False

Else

If Val(cboPageNo.Text) = 1 Then

cmdPrevPage.Enabled = False

cmdNextPage.Enabled = True

ElseIf Val(cboPageNo.Text) = mTotalPages Then

cmdPrevPage.Enabled = True

cmdNextPage.Enabled = False

Else

cmdPrevPage.Enabled = True

cmdNextPage.Enabled = True

End If

End If

End Sub

Private Sub HScroll1\_Change()

Select Case mSizeNo

Case 3

Pic3.Left = -HScroll1.Value

'Case 5

' Pic5.Left = -HScroll1.Value

End Select

End Sub

Private Sub VScroll1\_Change()

Select Case mSizeNo

Case 3

Pic3.Top = -VScroll1.Value

'Case 5

' Pic5.Top = -VScroll1.Value

End Select

End Sub

Private Sub ChangePreview()

Select Case mSizeNo

Case 1

If mNotShow = False Then

Pic1.Visible = True

Else

Pic1.Visible = False

End If

Pic2.Visible = False

Pic3.Visible = False

Pic4.Visible = False

Pic5.Visible = False

Case 2

Pic1.Visible = False

If mNotShow = False Then

Pic1.Visible = True

Else

Pic2.Visible = False

End If

Pic2.Visible = True

Pic3.Visible = False

Pic4.Visible = False

Pic5.Visible = False

Case 3

Pic1.Visible = False

Pic2.Visible = False

If mNotShow = False Then

Pic3.Visible = True

Else

Pic3.Visible = False

End If

Pic4.Visible = False

Pic5.Visible = False

Case 4

Pic1.Visible = False

Pic2.Visible = False

Pic3.Visible = False

If mNotShow = False Then

Pic4.Visible = True

Else

Pic4.Visible = False

End If

Pic5.Visible = False

Case 5

Pic1.Visible = False

Pic2.Visible = False

Pic3.Visible = False

Pic4.Visible = False

If mNotShow = False Then

Pic5.Visible = True

Else

Pic5.Visible = False

End If

End Select

End Sub

' Combo does not honour "Change", we use "Click" instead

Private Sub cboScale\_Click()

Select Case cboScale.Text

Case "Zoom in"

mSizeNo = 3

Case "Zoom out"

mSizeNo = 5

End Select

If mSizeNo > 1 And mSizeNo < 5 Then

End If

ChangePreview

End Sub

Private Sub cmdPrint\_Click()

'PrintRTF frmMain1.RTBox, 720, 720, 720, 720 'Call PrintRTF sub

gprint = True

Unload Me

End Sub

Private Function MakeSizes(ByVal mofSize As Integer) As Boolean

Dim SrcX As Long, SrcY As Long

Dim DestX As Long, DestY As Long

Dim SrcWidth As Long, SrcHeight As Long

Dim DestWidth As Long, DestHeight As Long

Dim SrcHDC As Long, DestHDC As Long

Dim mresult

SrcX = 0: SrcY = 0: DestX = 0: DestY = 0

SrcWidth = PicX.ScaleWidth

SrcHeight = PicX.ScaleHeight

SrcHDC = PicX.hdc

Select Case mofSize

Case 1

DestWidth = Pic1.ScaleWidth

DestHeight = Pic1.ScaleHeight

DestHDC = Pic1.hdc

Case 2

DestWidth = Pic2.ScaleWidth

DestHeight = Pic2.ScaleHeight

DestHDC = Pic2.hdc

Case 3

DestWidth = Pic3.ScaleWidth

DestHeight = Pic3.ScaleHeight

DestHDC = Pic3.hdc

Case 4

DestWidth = Pic4.ScaleWidth

DestHeight = Pic4.ScaleHeight

DestHDC = Pic4.hdc

Case 5

DestWidth = Pic5.ScaleWidth

DestHeight = Pic5.ScaleHeight

DestHDC = Pic5.hdc

End Select

mresult = StretchBlt(DestHDC, DestX, DestY, DestWidth, DestHeight, SrcHDC, \_

SrcX, SrcY, SrcWidth, SrcHeight, SRCCOPY)

If mresult = 0 Then

MsgBox "Error occurred in sizing images. Cannot continue"

MakeSizes = False

Else

MakeSizes = True

End If

End Function

Private Sub cmdClose\_Click()

Unload Me

End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

' To display the same as it would print on the selected printer

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function DocWYSIWYG(RTB As Control) As Long

Dim LeftMargin As Long, RightMargin As Long

Dim LeftOffset As Long, TopOffset As Long

Dim linewidth As Long

Dim PrinterhDC As Long

Dim r As Long

Dim LeftMarginWidth

Dim RightMarginWidth

Dim TopMarginHeight

Dim BottomMarginHeight

LeftMarginWidth = 720

TopMarginHeight = 720

RightMarginWidth = 720

BottomMarginHeight = 720

Printer.Print Space(1)

Printer.ScaleMode = vbTwips

' Get the offsett to the printable area on the page in twips

LeftOffset = Printer.ScaleX(GetDeviceCaps(Printer.hdc, PHYSICALOFFSETX), vbPixels, vbTwips)

TopOffset = Printer.ScaleY(GetDeviceCaps(Printer.hdc, PHYSICALOFFSETY), vbPixels, vbTwips)

' Calculate the Left, Top, Right, and Bottom margins

LeftMargin = LeftMarginWidth - LeftOffset

TopMargin = TopMarginHeight - TopOffset

RightMargin = (Printer.Width - RightMarginWidth) - LeftOffset

BottomMargin = (Printer.Height - BottomMarginHeight) - TopOffset

' LeftMargin = gLeftMargin \* 1440

'RightMargin = Printer.Width - gRightMargin \* 1440

linewidth = RightMargin - LeftMargin

DocWYSIWYG = linewidth

End Function

Sub FormPreviewPage(inControl As Control, InPage As Integer)

Dim PageCtn

' Clear picture box control

Set inControl.Picture = LoadPicture

' Set printable area rect.

' Note in preview, scaleModes are all in vbPixels,

' have to compute the vbtwips equivalent

' rectPage.Left = 0

'rectPage.Top = 0

'rectPage.Right = inControl.Width \* Screen.TwipsPerPixelX

'rectPage.Bottom = inControl.Height \* Screen.TwipsPerPixelY

rectPage.Left = 0

rectPage.Top = 0

rectPage.Right = Printer.ScaleWidth

rectPage.Bottom = Printer.ScaleHeight

LeftOffset = Printer.ScaleX(GetDeviceCaps(inControl.hdc, PHYSICALOFFSETX), vbPixels, vbTwips)

TopOffset = Printer.ScaleY(GetDeviceCaps(inControl.hdc, PHYSICALOFFSETY), vbPixels, vbTwips)

' Calculate the Left, Top, Right, and Bottom margins

LeftMargin = LeftMarginWidth - LeftOffset

TopMargin = TopMarginHeight - TopOffset

RightMargin = (Printer.Width - RightMarginWidth) - LeftOffset

BottomMargin = (Printer.Height - BottomMarginHeight) - TopOffset

rectDrawTo.Left = LeftMargin

rectDrawTo.Top = TopMargin

rectDrawTo.Right = RightMargin

rectDrawTo.Bottom = BottomMargin

' Set rect in which to print (relative to printable area)

' rectDrawTo.Left = gLeftMargin \* 1440

'rectDrawTo.Top = gTopMargin \* 1440

'rectDrawTo.Right = inControl.Width \* Screen.TwipsPerPixelX \_

- gRightMargin \* 1440

'rectDrawTo.Bottom = inControl.Height \* Screen.TwipsPerPixelY \_

- gBottomMargin \* 1440

mFormatRange.hdc = inControl.hdc ' Use the same DC for measuring and rendering

mFormatRange.hdcTarget = inControl.hdc ' Point at hDC

mFormatRange.rectRegion = rectDrawTo ' Area on page to draw to

mFormatRange.rectPage = rectPage ' Entire size of page

mFormatRange.mCharRange.firstChar = 0 ' Start of text

mFormatRange.mCharRange.lastChar = -1 ' End of the text

TextLength = Len(frmmain1.RTBox.Text)

PageCtn = 1

Do

newStartPos = SendMessage(frmmain1.RTBox.hwnd, EM\_FORMATRANGE, True, mFormatRange)

If newStartPos >= TextLength Then

Exit Do

End If

If PageCtn = InPage Then

Exit Do

End If

' Clear picture box control

Set inControl.Picture = LoadPicture

mFormatRange.mCharRange.firstChar = newStartPos ' Starting position for next page

mFormatRange.hdc = inControl.hdc

mFormatRange.hdcTarget = inControl.hdc

PageCtn = PageCtn + 1

DoEvents

Loop

dumpaway = SendMessage(inControl.hwnd, EM\_FORMATRANGE, False, ByVal CLng(0))

End Sub

' Test how many pages are there in total

Function PageCtnProc(inControl As Control) As Integer

Dim mPageCtn As Integer

' Set printable area rect.

' Note in preview, scaleModes are all in vbPixels;

' convert them to vbtwips.

' rectPage.Left = 0

' rectPage.Top = 0

' rectPage.Right = inControl.Width \* Screen.TwipsPerPixelX

' rectPage.Bottom = inControl.Height \* Screen.TwipsPerPixelY

' Set rect in which to print (relative to printable area)

' rectDrawTo.Left = gLeftMargin \* 1440

' rectDrawTo.Top = gTopMargin \* 1440

' rectDrawTo.Right = inControl.Width \* Screen.TwipsPerPixelX \_

- gRightMargin \* 1440

' rectDrawTo.Bottom = inControl.Height \* Screen.TwipsPerPixelY \_

- gBottomMargin \* 1440

' Set up the print instructions

LeftOffset = Printer.ScaleX(GetDeviceCaps(Printer.hdc, PHYSICALOFFSETX), vbPixels, vbTwips)

TopOffset = Printer.ScaleY(GetDeviceCaps(Printer.hdc, PHYSICALOFFSETY), vbPixels, vbTwips)

' Calculate the Left, Top, Right, and Bottom margins

LeftMargin = LeftMarginWidth - LeftOffset

TopMargin = TopMarginHeight - TopOffset

RightMargin = (Printer.Width - RightMarginWidth) - LeftOffset

BottomMargin = (Printer.Height - BottomMarginHeight) - TopOffset

rectDrawTo.Left = LeftMargin

rectDrawTo.Top = TopMargin

rectDrawTo.Right = RightMargin

rectDrawTo.Bottom = BottomMargin

' Set rect in which to print (relative to printable area)

' rectDrawTo.Left = gLeftMargin \* 1440

'rectDrawTo.Top = gTopMargin \* 1440

'rectDrawTo.Right = inControl.Width \* Screen.TwipsPerPixelX \_

- gRightMargin \* 1440

'rectDrawTo.Bottom = inControl.Height \* Screen.TwipsPerPixelY \_

- gBottomMargin \* 1440

mFormatRange.hdc = inControl.hdc ' Use the same DC for measuring and rendering

mFormatRange.hdcTarget = inControl.hdc ' Point at hDC

mFormatRange.rectRegion = rectDrawTo ' Area on page to draw to

mFormatRange.rectPage = rectPage ' Entire size of page

mFormatRange.mCharRange.firstChar = 0 ' Start of text

mFormatRange.mCharRange.lastChar = -1 ' End of the text

mFormatRange.hdc = inControl.hdc ' Use the same DC for measuring and rendering

mFormatRange.hdcTarget = inControl.hdc ' Point at hDC

mFormatRange.rectRegion = rectDrawTo ' Area on page to draw to

mFormatRange.rectPage = rectPage ' Entire size of page

mFormatRange.mCharRange.firstChar = 0 ' Start of text

mFormatRange.mCharRange.lastChar = -1 ' End of the text

TextLength = Len(frmmain1.RTBox.Text)

mPageCtn = 1

Do

' Print the page by sending EM\_FORMATRANGE message

newStartPos = SendMessage(frmmain1.RTBox.hwnd, EM\_FORMATRANGE, True, mFormatRange)

If newStartPos >= TextLength Then

Exit Do

End If

mFormatRange.mCharRange.firstChar = newStartPos ' Starting position for next page

mFormatRange.hdc = inControl.hdc

mFormatRange.hdcTarget = inControl.hdc

mPageCtn = mPageCtn + 1

DoEvents

Loop

' Clear picture box control

Set inControl.Picture = LoadPicture

dumpaway = SendMessage(inControl.hwnd, EM\_FORMATRANGE, False, ByVal CLng(0))

PageCtnProc = mPageCtn

End Function

Sub DocPrintProc()

On Error Resume Next

DoEvents

' Clear picture box control

Set preview.picP.Picture = LoadPicture

Dim mydialog1 As Object

Dim mFromPage As Integer, mToPage As Integer, mpage As Integer

Set mydialog1 = frmmain1.dlgCommonDialog

mydialog1.DialogTitle = "Print"

mydialog1.CancelError = True

' Allow user select page range

mydialog1.FLAGS = cdlPDReturnDC + cdlPDPageNums

' But default to one of these

If frmmain1.RTBox.SelLength = 0 Then

mydialog1.FLAGS = mydialog1.FLAGS + cdlPDAllPages

Else

mydialog1.FLAGS = mydialog1.FLAGS + cdlPDSelection

End If

mydialog1.ShowPrinter

If Err = MSComDlg.cdlCancel Then

Exit Sub

End If

mFromPage = mydialog1.FromPage

mToPage = mydialog1.ToPage

'If frmFrame.ActiveForm.WindowState <> 1 Then

'DocWYSIWYG frmMain1.RTBox

'frmFrame.ActiveForm.Move 0, 0

'Else

' MsgBox "Cannot proceed with minimized screen"

'Exit Sub

'End If

'If MsgBox("Proceed to print", vbYesNo + vbQuestion) = vbNo Then

' Exit Sub

'End If

Printer.Print ""

Printer.ScaleMode = vbTwips

' Set printable rect area

rectPage.Left = 0

rectPage.Top = 0

rectPage.Right = Printer.ScaleWidth

rectPage.Bottom = Printer.ScaleHeight

' Set rect in which to print (relative to printable area)

rectDrawTo.Left = gLeftMargin \* 1440

rectDrawTo.Top = gTopMargin \* 1440

rectDrawTo.Right = Printer.ScaleWidth - gRightMargin \* 1440

rectDrawTo.Bottom = Printer.ScaleHeight - gBottomMargin \* 1440

' Dump earlier pages if any to PicP before reaching first wanted page

mFormatRange.hdc = preview.picP.hdc

mFormatRange.hdcTarget = preview.picP.hdc

newStartPos = 0 ' Next char to start

mFormatRange.rectRegion = rectDrawTo ' Area on page to draw to

mFormatRange.rectPage = rectPage ' Entire size of page

mFormatRange.mCharRange.firstChar = newStartPos ' Start of text

mFormatRange.mCharRange.lastChar = -1 ' End of the text

TextLength = Len(frmmain1.RTBox.Text)

' Dumping if any

mpage = 1

Do

If mpage = mFromPage Then

Exit Do

End If

' Don't clear picture box control here, unless you want to print

' from first page always.

' Print the page by sending EM\_FORMATRANGE message

newStartPos = SendMessage(frmmain1.RTBox.hwnd, EM\_FORMATRANGE, True, mFormatRange)

If newStartPos >= TextLength Then

Exit Do

End If

mFormatRange.mCharRange.firstChar = newStartPos ' Starting position for next page

mFormatRange.hdc = preview.picP.hdc

mFormatRange.hdcTarget = preview.picP.hdc

mpage = mpage + 1

DoEvents

Loop

' Must cleanse memory here before print, otherwise font will not be right

'dumpaway = SendMessage(Screen.frmMain1.RTBox.hwnd, EM\_FORMATRANGE, False, ByVal CLng(0))

If newStartPos >= TextLength Then

Exit Sub

End If

' Have to reinitialize printer here

Printer.Print ""

Printer.ScaleMode = vbTwips

' Actual print to printer, starting from the user-selected Page No.

mFormatRange.hdc = Printer.hdc

mFormatRange.hdcTarget = Printer.hdc

' Update char range

mFormatRange.mCharRange.firstChar = newStartPos

Do

' Print the page by sending EM\_FORMATRANGE message

newStartPos = SendMessage(frmmain1.RTBox.hwnd, EM\_FORMATRANGE, True, mFormatRange)

If newStartPos >= TextLength Then

Exit Do

End If

If mpage = mToPage Then

Exit Do

End If

mFormatRange.mCharRange.firstChar = newStartPos ' Starting position for next page

Printer.NewPage ' Move on to next page

Printer.Print "" ' Re-initialize hDC

mFormatRange.hdc = Printer.hdc

mFormatRange.hdcTarget = Printer.hdc

mpage = mpage + 1

DoEvents

Loop

' Commit the print job

Printer.EndDoc

' Free up memory

'dumpaway = SendMessage(Screen.frmMain1.RTBox.hwnd, EM\_FORMATRANGE, False, ByVal CLng(0))

End Sub