What about plexityHide component printing?

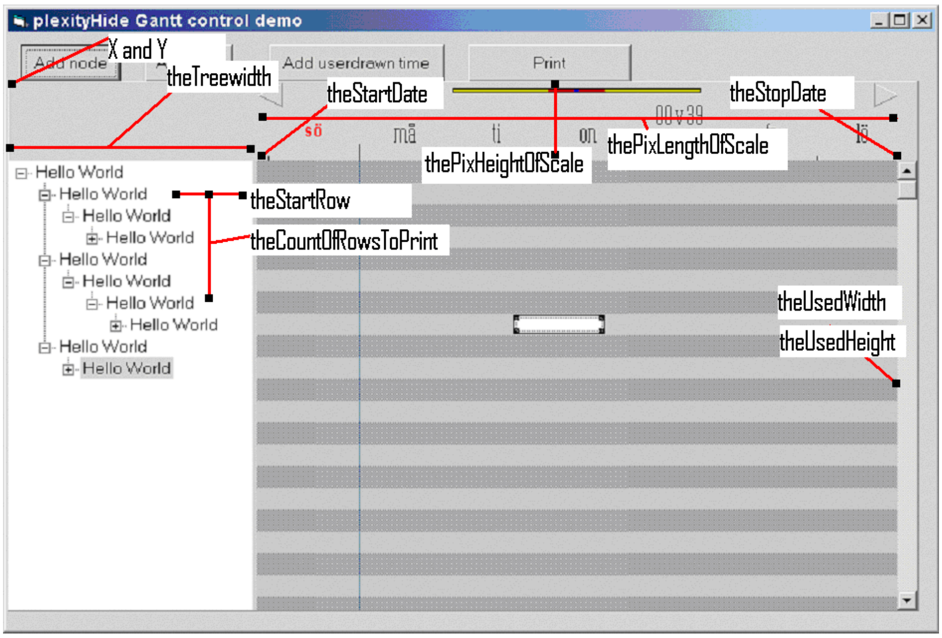
Usually you would expect a method named print on the component, and this method would read your mind and present a perfect view on paper on your favourite printer. But then maybe the control reads your mind wrong and does not do exactly as you expect, this is a cruel awakening after the first fascination of paper printouts has faded away. What about a preview? What about integration with my favourite report tool? What about dividing the printout over multiple sheets of paper of A5-size?

How can we as a tool provider keep everyone happy? Our answer is to leave it up to you! Just provide the necessary tool with enough parameters to effect the layout of the print.

How?
All drawing in windows are done in a device context, be it a printer or a monitor, so provide a tool that can output to any given device context. This might sound perfect, but the drawback is that there are quite a few parameters to set.

Printing the phGantx

```
procedure PrintToHdc(aHDC: Integer;
                     X: Integer; Y: Integer;
                     xScale: Double; yScale: Double;
                     theTreewidth: Integer;
                     const theStartRow: IphDataEntity_Tree;
                     theCountOfRowsToPrint: Integer;
                     theExpandAll: WordBool;
                     theStartDate: TDateTime; theStopDate: TDateTime;
                     thePixLengthOfScale: Integer; thePixHeightOfScale: Integer;
                     var theUsedWidth: Integer; var theUsedHeight: Integer);
```



In the picture above most of the parameters are shown. All integers are in pixels ,except hdc which is a handle of a device context, and theCountOfRowsToPrint which is the number of rows in the tree that you want to print.

aHDC	Integer	Handle to device context	Can be a handle of printer, or why not a bitmap used to render jpeg files for html exposure?
X	Integer	Pixel of x start point on paper (context) of printout rectangle	
Y	Integer	Pixel of y start point on paper of printout rectangle	
Xscale	Double	All X sizes are multiplied with the X scale	
Yscale	Double	All Y sizes are multiplied with the Y scale	
TheTreewidth	Integer	The width of the tree in pixels (before X scaling)	
Const theStartRow	IphDataEntity_Tree	The tree node to start with	Nil means first
TheCountOfRowsToPrint	Integer	The integer count of rows to print	-1 means “all”
TheExpandAll	WordBool	When true all nodes are expanded in the printout	If false all nodes are left as they where
TheStartDate	TDateTime	The start date of the scale	
TheStopDate	TDateTime	The stop date of the scale	

ThePixLengthOfScale	Integer	The length of pixels in the scale before X scaling	
ThePixHeightOfScale	Integer	The height in pixels of the scale before Y scaling	
theUsedWidth	Integer	The calculated amount of X pixels used for rectangle printout	Strictly out parameter
var theUsedHeight	Integer	The calculated amount of Y pixels used for rectangle printout	Strictly out parameter

Now is the question about getting a hold of a device context. May we suggest that you try with a preview before starting to use paper:

```
Private Sub Command2_Click()
  Dim w As Long
  Dim h As Long

  phGantX1.PrintToHdc Picture1.hdc, 1, 1, 0.3, 0.3, 100, phGantX1.TopItemTree, -1,
True, Now - 30, Now + 30, 300, 60, w, h

End Sub
```

Then you can use the standard printer in VB:

```
Private Sub Command1_Click()
Dim w As Long
Dim h As Long
Dim w2 As Long
Dim h2 As Long


Printer.Print "This is a test printout, using three prints to look like one"
phGantX1.PrintToHdc Printer.hdc, 100, 100, 5, 5, 200, phGantX1.TopItemTree, -1,
True, Now - 30, Now + 30, 300, 30, w, h
phGantX1.PrintToHdc Printer.hdc, 100 + w, 100, 5, 5, 0, phGantX1.TopItemTree, -1,
True, Now + 30, Now + 90, 300, 30, w2, h2
phGantX1.PrintToHdc Printer.hdc, 100, 100 + h, 5, 5, 200, phGantX1.TopItemTree, -
1, True, Now - 30, Now + 30, 300, 0, w2, h2
Printer.EndDoc
End Sub
```