Xbase++ GraPDF Class

User guide
2020

Created by Softsupply Informatica

Rua Alagoas, 48
01242-000
São Paulo, SP
Brazil
Tel (5511) 3159-1997
Email : softsupply@terra.com.br

Contact : Edgar Borger

Table of contents

Intro 3
Class Methods 4
Document 5
New 6
Create 7
Page 8
Preview 10
CreatePDF 11
PagePrint 12
Save 13
Load 14
Text and Fonts 15
Font 16
FontDef 17
StringAt 18
mText 19
rText 21
TextColor 22
ChineseFonts 23
Drawing 24
Line 25
Box 26
Shape 27
Image 28
LineColor 29
BoxColor 30
LineWidth 31
Charts 32
BarChart 33
LineChart 34
PieChart 35
XbasePDF 36
PDF 37
Class Variables 38
Notes 39
Demo 40

Intro

This class was created to help Xbase++ users in creating reports with the ability to preview on screen, print and convert to PDF from Xbase++ applications. This solution can be deployed to customer installations without further steps, like PDF printer drivers, and so on.

This is a very simple class with very little effort in terms of programming, of course implementing it will require code changes, but the benefits are worth it.

An example of a simple program would be :

```
PROCEDURE MAIN
    Grapdf := xbpGraPdf():New()          // Creates the GraPDF object

    Grapdf:Create( oPrinter )            // Tell the class where should the report be printed

    Grapdf:Page()                        // Page size

    Grapdf:Font( '10.Courier New' )      // Choose font

    Grapdf:StringAt( {10, 10}, "Hello
    World !" )                           // write text

    Grapdf:Preview()                     // Preview the report

return
```

Class Methods

Document

Those are the document handling methods.

New

- **New()** => Creates the Grapdf object. No parameters are needed, and returns the new created object.

Create

- **Create(** oPrinter **)** => Declares the printer object where the report should be printed, if omitted, the default Windows printer will be used.

  oPrinter (object) : is the printer object for printing.

  Example : oPrt := xbpPrinter():New():Create()
  GraPdf:Create(oPrt)

Page

- **Page(** *[page size], [orientation], [lines], [columns], [user size], [Coord System], [left margin], [top margin]* **)**
  => starts a new page, specifying the page size, orientation and the coordinates system used.

Letter	230 x 275	8 1/2 x 11
LetterSmall	230 x 279	8 1/2 x 11
Tabloid	279 x 432	11 x 17
Ledger	432 x 279	17 x 11
Legal	203 x 356	8 1/2 x 14
Statement	127 x 203	5 1/2 x 8 1/2
Executive	184 x 254	7 1/4 x 10 1/2
A3	297 x 420	
A4	210 x 297	
A4 Small	210 x 297	
A5	148 x 210	
B4	250 x 354	
B5	182 x 257	
Folio	203 x 330	8 1/2 x 13
Quarto	215 x 275	
10 x 14 in	254 x 356	10 x 14
11 x 17 in	279 x 432	11 x 17
Note	203 x 279	8 1/2 x 11
Envelope #9	98 x 225	3 7/8 x 8 7/8
Envelope #10	105 x 229	4 1/8 x 9 1/2
Envelope #11	102 x 264	4 1/2 x 10 3/8
Envelope #12	102 x 179	4 3/4 x 11
Envelope #14	127 x 279	5 x 11 1/2
C size sheet	432 x 558	17 x 22
D size sheet	558 x 864	22 x 34
E size sheet	864 x 1118	34 x 44
DL	110 x 220	
C5	162 x 229	
C3	324 x 458	
C4	229 x 324	
C6	114 x 162	
C65	114 x 229	
B4	250 x 353	
B5	176 x 250	
B6	176 x 125	

Envelope Italy	110 x 230	
Monarch	98 x 191	3 7/8 x 7 1/2
Envelope Personal	92 x 165	3 5/8 x 6 1/2
US Standard Fanfold	378 x 279	14 7/8 x 11
German Standard Fanfold	216 x 305	8 1/2 x 12
German Legal Fanfold	216 x 330	8 1/2 x 13

Orientation (numeric) : is the page orientation, portrait or landscape, and defaults to XBPPRN_ORIENT_PORTRAIT.

Lines (numeric) : is the number of lines in the page, this number is related to the chosen coordinate system (see bellow). Default is 66 lines per page.

Columns (numeric) : the number of character columns in each line, this number is also related to the coordinate system. Default is 80 characters per line.

User size (array) : is a two element array that specifies page dimensions, first element is horizontal size, and second element is vertical size. This parameter is only used for *page size* XBPPRN_FORMAT_USER, and the measures should be in 1/72 inches. A page with 3 x 5 inches should be informed as { 216, 360 }.

Coordinate system (numeric): the coordinates system specifies the type of coordinates (line, column) passed to the class, they can be :

0 = COL/LINE	Column, Line, Top Down
2 = GRA_PU_LOMETRIC	0.1 Millimeters, Bottom Up
100 = CLIPPER	Line, Column, Top Down (Clipper style)

left margin (numeric) : left margin for the document, default is one character.

top margin (numeric) : top margin for the document, default is one line.

Examples : oPdf:NewPage(XBPPRN_FORMAT_A4) will create a page in the A4 format, oriented portrait, with 66 lines and 80 columns, using line/column coordinate system.

oPdf:NewPage(XBPPRN_FORMAT_LETTER, , 660, 800) will create portrait oriented page, letter size, with standard line/column coordinate system, but with 660 lines per page and 800 characters per line, creating a smaller line/character feed, that could provide better precision in positioning elements on the page.

oPdf:NewPage(XBPPRN_FORMAT_LETTER, , , , 3) this page will have letter size, but will be using HIMETRIC coordinate system.

oPdf:NewPage(XBPPRN_FORMAT_USER, , 30, 70, { 504, 360 }) this page will be sized at 5 x 7 inches providing 30 lines with 70 characters in each line, notice that the horizontal dimension (504) is larger that the vertical (360), indicating that the paper will be at a landscape position.

Preview

- **Preview(** *nZoom* **)** => finish all processing, and opens the preview window.

options are : *nZoom* : indicates the initial zoom for the preview window, default is 1.00 (100 %)
-1 : indicates initial zoom to height of window
-2 : indicates initial zoom to width of window
-3 : indicates initial zoom to hole window size (proportions may be distorted)

CreatePDF

- **CreatePDF(** *PDFName* **)** => Create the PDF file without showing the preview window. If no *PDFNname* is given, a dialog will ask for the name.

PagePrint

- **PagePrint( )** => prints the report to the designated printer, without showing the preview window. On return variable *lPrinted* will be set to TRUE.

Save

- **Save(** *cFileName* **)** => saves all GraPDF calls to a XPF file. This file can latter be retrieved with the Load call, restoring the current status for the report. Default file name is "GraPDF".

This process enables the creation of "templates" for your report, you can design a page, with all the boxes, images, and fixed text data, save it to a file, and then restore it back latter on, just adding variable data into fields.

Load

- **Load(** *cFileName* **)** => Loads all GraPDF calls from a XPF file taht has been previously created with the Save call, restoring the current status for the report. Default file name is "GraPDF".

Text and Fonts

Those are text methods, with them you can load any kind of text into your document.

Font

- **Font(** *[compound fontname], [Encoding], [Embed], [translate]* **)** => specify the wanted font according to Xbase syntax, nominal point size followed by dot followed by font name. If none provided defaults to “10.Courier New”, with WinAnsiEncoding. Basic fonts are included in the Xbase PDF Class, but with version 3, any True Type font can be used to create your document, (see FontDef bellow), also any size can be used, there is no limitation to maximum size or the number of fonts used in one document. The existing fonts can be combined with Bold and or Italic, and “Underscore” can be added to any combination, providing underlining of text.

  Compound fontname (character) : is the font compound name, point size, followed by font family name. Valid fonts are any font installed in the system, combined to Bold, Italic or Bold Italic. Underscore can be added to any of the available fonts, like Font(“10.Arial Underscore”) or Font(“12.Helvetica Bold Underscore”). Other fonts might be used using the **FontDef** method.

  Encoding (character) : This parameter can be used to specify different encoding for the chosen font during the creation of the PDF file, one of the following values should be used :

Encoding	**Comments**
CP1250	Central Europe
CP1251	Cyrillic
CP1253	Greek
CP1254	Turkish
CP1255	Hebrew
CP1256	Arabic
CP1257	Baltic
CP1258	Vietnam

  Embed (logical) : specify if the font file is to be embedded in the document, or not. Embedding the font makes the document larger in size, but assures that the font will be respected, if not embedded, the reader may chose a similar font for displaying the document. This option is valid only for the PDF document to be created.

  Examples : GraPdf:Font(“10.Arial”) will print with Arial fonts using 10 point size characters like
  => abcdefABCDEF12345.

FontDef

- **FontDef(** *family name, file name, embed* **)** => this powerful method enables you to use ant True Type font installed in your system. By defining the font family name, and its file container, you can use any existing font, for non Latin characters, bar codes, symbols, etc.

  Family name (character) : is the font compound name that you will be using latter to create your document.

  File name (character) : is the name of the file for that specified font. This file must be in the %SystemRoot %\Fonts folder or at the current working directory and must be a either a TTF format file, or a AFM/PFB file. If the file is not found at usage time, font will default to Courier New.

  Embed (logical) : indicates if the font should be embedded (incorporated) to the document, if TRUE, the TTF file will be embedded to the PDF file, making it larger in size, but with a better font definition, if FALSE, the font will no be incorporated to the file, making it smaller, but eventually having a substitute font if the same font is not installed at viewing time.

  Examples : GraPdf:FontDef(“3-9 Barcode”, “FREE3OF9X.TTF”) this will add the barcode TTF font to your pdf, enabling you to print barcode data.
  GraPdf:FontDef(“Avant Garde Book”, “avgardn.ttf”) will allow you to use the Avant Garde font in your document.

  Observation : you must define all the variations that you want to use in a font, there are separate files for Bold, Italic and Bold Italic fonts, and they must be defined one by one to XbpGRPDF.

StringAt

- **StringAt(** *{ xPos, yPos} , string* **)** => prints a string of text on the page, at the position indicated by xPos and yPos coordinates.

  yPos (numeric) : is the vertical position where the text will be outputted, this position must be according to the coordinate system chosen for your document (see NewPage above). If *line* is NIL, will continue to print at current line

  xPos (numeric) : is the horizontal position within the page for the text, according to the coordinate system specified in the NewPage method. If *column* is NIL, will print at the next character position.

  String (character) : is the string of characters to be outputted at the specified position, if the string does not fit on the line, it will be lost, no line wrapping is done with this method, for this use the mText method.

  Examples : GraPdf:StringAt({50, 10}, “this is a sample”) with print “this is a sample” on position { 50, 10 } of the current page.

Obs: the variable GraPDF:Fixed will signal the class to use a fixed width for all characters displayed with this method. If this variable is set to "true" than the characters will be evenly displayed along the line, if "false" than characters will be displayed according to the chosen font.

---

mText

- **Mtext(** *{ xpos, yPos }, text, horizontal alignment, width* **)** => Multi line text, automatically fits the given text into the specified coordinates, aligning the text horizontally relative to *xPos*.

  yPos (numeric) : specifies the vertical position according to the coordinate system for the document, where text output will begin.

  xPos (numeric) : specifies the horizontal position for the text output. This parameters is used by XbpGraPDF according to the *horizontal alignment* chosen. On *left* alignment (standard) and on *justified* the left margin of the text is aligned to the specified column, on *right* alignment the end of the text lines are aligned to the column, and on *center* the text is centralized on the column position.

  Text (characters) : is the text to be outputted at the document, this text could be several lines of text, separated by NL ($0D0A) characters, or in the case of *justified* alignment one long string (there is no limit). Text should conform to memo fields formats.

  Horizontal alignment (numeric) : is the type of alignment to be performed by XbpGraPDF on adding the text to your document.

  Possible values for horizontal alignment are :

Left	=> 0 (Default)
Right	=> 1
Center	=> 2
Justified	=> 3

  Width (numeric) : on *justified* the optional parameter *width* specifies the line width (in terms of characters), if this parameter is not present, the width will be of the longest line present at the text.

  Examples : for this example the text will be “This is a sample” (one word on each line)

```
Text := “This” + nl + “is” + nl + “a” + “Sample”
GraPdf:Box( {10, 0}, {18, 20} )
GraPdf:Line( {14, 0}, {14, 20} )
GraPdf:Line( {10, 10}, {14. 10} )

GraPdf:mText( {10, 10}, text, 0 )        (1)
GraPdf:mText( {10, 10}, text, 1 )        (2)
GraPdf:mText( {14, 10}, text, 2 )        (3)
GraPdf:mText( {18, 10}, text, 3, 16 )    (4)
```

2	This Is A Sample	This Is A Sample	1
3	This Is A Sample		
4		This Is a Sample of Justified Text	

Notice that all mText calls use column 10, so the positioning of the text varies according to the chosen alignment.

rText

- **RText(** *line, column, string, angle* **)** => prints a string of text on the page, rotating to the specified *angle*, at line and column position that are calculated according to page size.

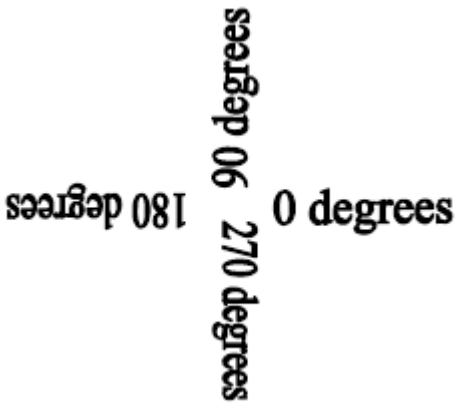
Line (numeric) : Vertical position of text according to coordinate system.

Column (numeric) : Horizontal position of text according to coordinate system.

String (character) : Text to be included in the document.

Angle (numeric) : Angle of rotation, ranging from 0 to 360, starting at the normal text position and rotating counter clockwise back to an horizontal position.

Examples : GraPdf:rText(15, 15, “normal”, 0)
GraPdf:rText(15, 25, “90 degrees”, 90)
GraPdf:rText(15, 25, “180 degrees”, 180)
GraPdf:rText(15, 25, “270 degrees”, 270)



TextColor

- **TextColor(** [*color]* **)** => specify the color to the following text.

  color (numeric) : informs the color for the following text, constants from GRA.CH can be used in the form of GRA_CLR_… or any other RGB number (created with GraMakeRGBColor()) can be used. Default is GRA_CLR_BLACK.

ChineseFonts

- **ChineseFonts(** [en] **)** => enable the use of chinese MingLiU font.

  this command will make available the MingLiU chinese font with variations of bold and italic, the usage should be pdf:Font("anysize.MingLiU"), or "MingLiU,Bold", "MingLiU,Italic" and "MingLiU,BoldItalic" using "," (comma) instead of space between the font name and the variation.

Drawing

This group of methods can be used to load drawing and images to your document.

Line

- **Line(** *{ init xPos, init yPos}, {final xPos, final yPos* **)** => draws a line from initial position to final position.

  Init yPos (numeric) : Vertical position of line start according to coordinate system.

  Init xPos (numeric) : Horizontal position to start drawing the line.

  Final yPos (numeric) : Vertical position of line ending.

  Final xPos (numeric) : Horizontal position of line ending.

Box

- **Box(** *{init xPos, init yPos}, {final xPos, final yPos}, [fill percent]* **)** => draws a box (rectangle) from initial {x,y} position to final {x,y} position. When creating a PDF document, optionally fills the created box with gray according to fill percent value.

  Init yPos (numeric) : Vertical position of box corner according to coordinate system.

  Init xPos (numeric) : Horizontal position of box corner to start drawing the box.

  Final yPos (numeric) : Vertical position of opposite box corner.

  Final xPos (numeric) : Horizontal position of opposite box corner.

  Fill percent (numeric) : numeric value in the range of 0 to 100, indicating the amount of gray that should be used to fill the box, the higher the number the darker will be the filling, a value between 5 to 20 will provide a gray area as a background, suitable for foreground text, or other data. Default for fill percent is 0 (none).

Shape

- **Shape(** *points array, line color, [fill percent]* **)** => creates a geometric shape, defined by and array of points {{l1,c1},{l2,c2}…..}, drawn in the specified color (default is black), and optionally fills the create shape with gray according to fill percent value.

  Points array (array) : An array of numbers pairs, giving vertical and horizontal positions, for every point in the shape, the quantity of pairs is unlimited, allowing the creation of complex figures of any kind.

  Line color (numeric) : informs the color of the surrounding line for the shape, constants from GRA.CH can be used in the form of GRA_CLR_… or any other RGB number (created with GraMakeRGBColor()) can be used.

  Fill percent (numeric) : numeric value in the range of 0 to 100, indicating the amount of gray that should be used to fill the box, the higher the number the darker will be the filling, a value between 5 to 20 will provide a gray area as a background, suitable for foreground text, or other data. Default for fill percent is 0 (none).

  Example : GraPdf:Shape({ {10, 10}, {5, 15}, {10, 20} }, GRA_CLR_BLUE) will create a blue triangle at the specified points.

Image

- **Image(** *{init xPos, init yPos} {final xPos, final yPos}, image name, [ratio]* **)** => draws an image from initial line, initial column position to final line, final column position. Ratio indicates if the original ratio of the image is to be kept, or if the image should fill the entire area. Image can be JPEG, TIFF, BMP, PNG or GIF, file, and extension is mandatory.

  Init yPos (numeric) : Vertical position of image corner according to coordinate system.

  Init xPos (numeric) : Horizontal position of image corner to start drawing the box.

  Final yPos (numeric) : Vertical position of opposite image corner.

  Final xPos (numeric) : Horizontal position of opposite image corner.

  Image name (character) : Name of the image file to be included in the document, if the file is in a different folder as the current one, the full path should be given. Also the file type (JPEG, TIFF, BMP, PNG, or GIF) should be included in the name.

  Ratio (logical) : Ratio is true or false (.t. or .f.) indicating if the original ratio of the image is to be kept, or if the image should fill the entire area. If ratio is true, than *final line* and *final col* are used to calculate the positioning and sizing of the image, respecting however the original proportion.

LineColor

- **LineColor(** *[color]* **)** => specify the color to the following lines.

  Line color (numeric) : informs the color of the following lines, constants from GRA.CH can be used in the form of GRA_CLR_… or any other RGB number (created with GraMakeRGBColor()) can be used. Default is GRA_CLR_BLACK.

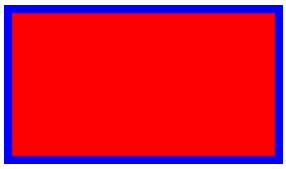
BoxColor

- **BoxColor(** *[color],[fill color]* **)** => specify the colors to the following box. First parameter is color for the box lines, and second parameter is color for filling the box (background).

  color (numeric) : informs the color drawing the following boxes, constants from GRA.CH can be used in the form of GRA_CLR_… or any other RGB number (created with GraMakeRGBColor()) can be used. Default is GRA_CLR_BLACK.

  Fill color (numeric) : informs the color for the filling of boxes, constants from GRA.CH can be used in the form of GRA_CLR_… or any other RGB number (created with GraMakeRGBColor()) can be used. Default is GRA_CLR_WHITE (no filling).

  Examples : GraPdf:BoxColor(GRA_CLR_BLUE, GRA_CLR_RED) followed by a oPdf:Box() call, will draw the a blue box with a red inside.



LineWidth

- **LineWidth(** *[width]* **)** => the width of the following lines.

  Width (numeric) : the width for the lines, starting with 1 and up, the higher the number, the thicker the line. Default is 1.

Charts

Those are Chart drawing methods. They can be used to create simple charts in your document, and are build using Box Lines, Shapes and Text methods.

BarChart

- **BarChart(** *{xPos, yPos}, {xSize, ySize}, data, data titles, chart title, color1, color2* **)** => draws a bar chart, at position *init line, init col* with size *height, width*, using the values in *data* array, for each bar, a title in *data titles* is displayed. Bars are drawn using *color1* for outline, and *color2* for filling. A title can be displayed at the top/ center of the chart using the *chart title* parameter. See a sample at Charts.PDF

  yPos (numeric) : Vertical position of chart corner according to coordinate system.

  xPos (numeric) : Horizontal position of corner to start drawing the chart.

  ySize (numeric) : Height of box containing the chart.

  xSize (numeric) : Width of box containing the chart.

  Data (array) : An array of data values to be charted, one bar will be drawn for each element of the array, automatic scaling will be calculated according to lowest/highest value present in the array.

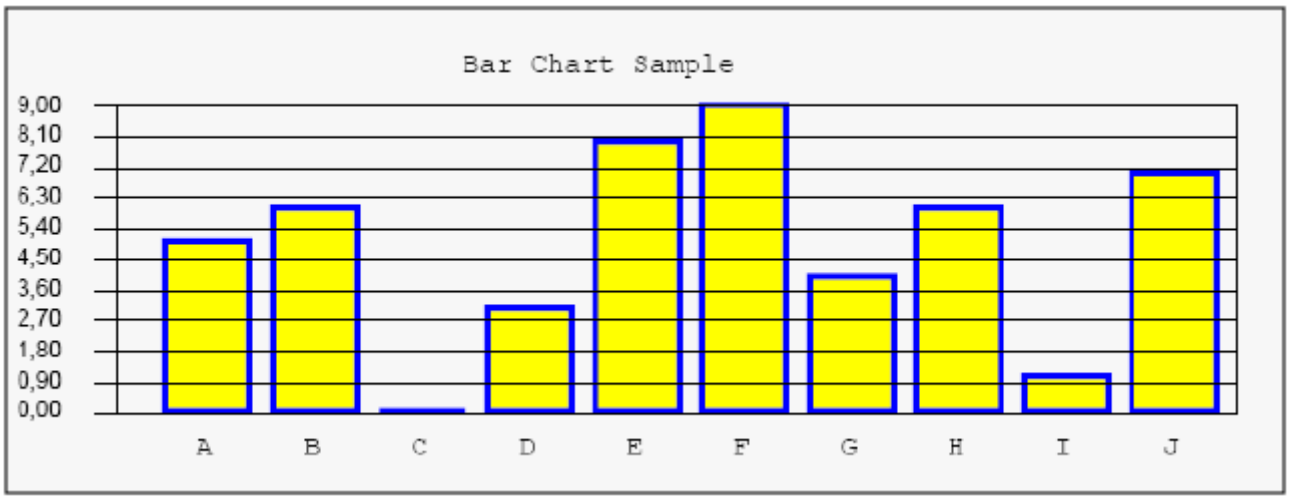
  Data titles (array) : An array containing titles for each element in the data array, those titles will be displayed under each bar, so the quantity of characters in each title should not be long, so as to fit under the bar.

  Chart title (character) : Title of the chart, will be displayed on the center of the chart top row.

  Color1 (numeric) : Color for the outline of the drawn bars. Conforms to color definition in GRA.CH or any valid RGB color.

  Color2 (numeric) : Color for the filling of the drawn bars. Conforms to color definition in GRA.CH or any valid RGB color.

  Example : oPdf:BarChart({100, 650}, {550, 240}, data, titu, "Bar Chart Sample", GRA_CLR_BLUE, GRA_CLR_YELLOW)
this will draw a bar chart with blue outlined bars and yellow fillings. See **Demo.PDF** for more details.



LineChart

- **LineChart(***{xPos, yPos}, {xSize, ySize}, data, data titles, chart title, colors* **)** => draws a line chart, at position *init line, init col* with size *height, width*, using the values in *data* array, for each position, a title in *data titles* is displayed. Multiple lines can be drawn, each one will have its color from the *colors* array. A title can be displayed at the top/center of the chart using the *chart title* parameter. See a sample at Charts.PDF

  yPos (numeric) : Vertical position of chart corner according to coordinate system.

  xPos (numeric) : Horizontal position of corner to start drawing the chart.

  ySize (numeric) : Height of box containing the chart.

  xSize (numeric) : Width of box containing the chart.

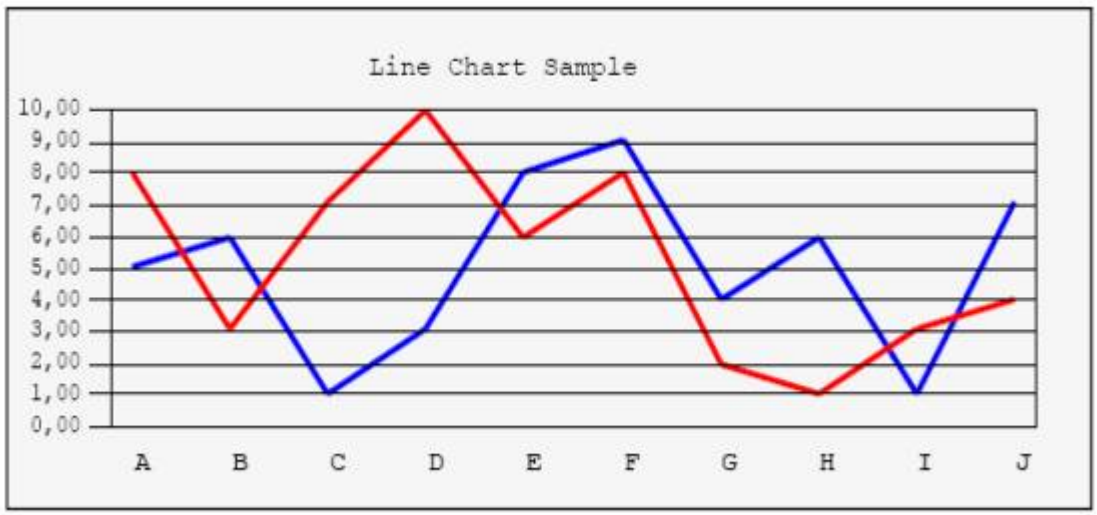
  Data (array) : An array of arrays. Each sub array contains data values to be charted, one line will be drawn for each element of the sub array, using different colors for each line, automatic scaling will be calculated according to lowest/highest value present in all the array.

  Data titles (array) : An array containing titles for each sub array in the data array, those titles will be displayed at the bottom of the chart.

  Chart title (character) : Title of the chart, will be displayed on the center of the chart top row.

  Colors (array) : An array of colors for the lines of the chart, should have as many elements as sub arrays in the data array. Conforms to color definition in GRA.CH or any valid RGB color.

  Example : oPdf:LineChart({100, 440}, {500, 240}, data, titu, "Line Chart Sample", { GRA_CLR_BLUE, GRA_CLR_RED })
this will draw a line chart with blue and red lines. See **Demo.PDF** for more details.



PieChart

- **PieChart(** *{xPos, yPos}, radius, data, data titles, chart title* **)** => draws a pie chart, at position *init line, init col* with size *radius*, using the values in *data* array, for each position, a title in *data titles* is displayed. Each pie "slice" will have its color. A title can be displayed at the top/center of the chart using the *chart* title parameter. See a sample at Charts.PDF

  yPos (numeric) : Vertical position of chart corner according to coordinate system.

  xPos (numeric) : Horizontal position of corner to start drawing the chart.

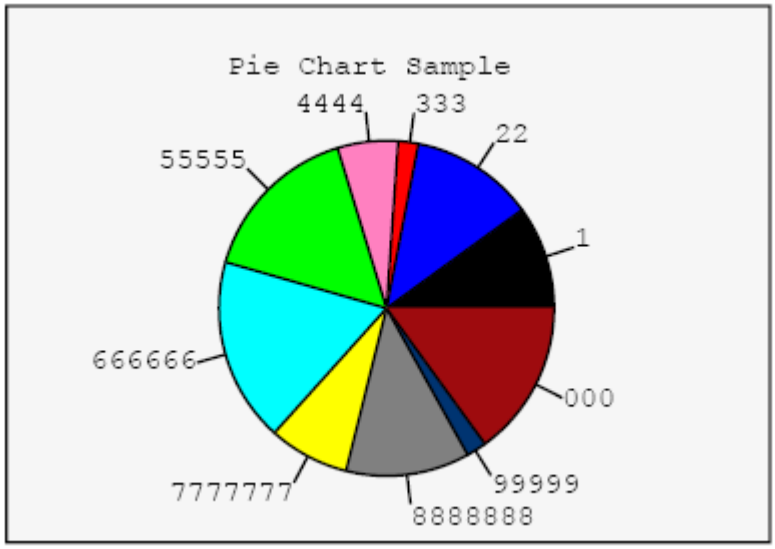
  Radius (numeric) : Height of box containing the chart.

  Data (array) : An array of data values to be charted, one pie slice will be drawn for each element of the array, using different colors for each slice.

  Data titles (array) : An array containing titles for each element in the data array, those titles will be displayed next to each slice in the chart.

  Chart title (character) : Title of the chart, will be displayed on the center of the chart top row.

  Example : oPdf:PieChart({300,1500}, 100, data, titu, "Pie Chart Sample")
  This will draw a pie chart. **See Demo.PDF** for more details.



XbasePDF

The GraPDF class interfaces with the XbasePDF class, allowing you to save your reports as a PDF file.

PDF

- **PDF(** *cPDFMethod, p1, p2, p3, p4, p5, p6, p7, p8, p9* **)** => calls a XbpPDF class method when creating a PDF file, this method has no effect on the GraPDF output, it only affects the resulting PDF.

  cPDFMethod : the method name to be passed to XbpPDF class when creating the PDF file.

  p1, ... p9 : up to nine parameters to be passed to the XbpPDF method.

Class Variables

aGraPDF *(array)*	array with contents of report, is the saved array when calling the "Save" method.
Title *(character)*	title for main preview window
PDFName *(character)*	file name for the pdf.
lPrinted *(logical)*	indicates if report was printed
lPDF *(logical)*	indicates if PDF file was created
Fixed *(logical)*	indicates fixed size width for characters. Default is "false"
cZoom *(character)*	zoom constant name. Default is "Zoom"
cPage *(character)*	page constant name. Default is "Page"
cTbar *(character)*	configures Tool Bar position, can be either "Top" or "Left". Default is "Top"

Notes

1. All color references are in the GraSetColor format and can be used from GRA.CH, additional colors can be created and used with GraMakeRGBColor() function.

2. Page Size and orientation can be used from XBPDEV.CH

3. Lines and Columns for all functions are according to the chosen coordinate system, in the Clipper/Text style (Coordinate System 0), the 0,0 point is top leftmost position, all others are Xbase style, meaning that the 0,0 point is at the bottom leftmost position of the page.

Demo

This is a sample program that is included in this package, by analyzing it, you will be able to see how simple its is, and at the same time how powerful this new class is.

```
#include "dll.ch"
#include "gra.ch"
#include "xbp.ch"
#include "common.ch"
#include "xbpdev.ch"
#pragma  library("xppPDF1.lib")
#pragma  library("xppGRPDF.lib")

PROCEDURE MAIN

    use parts

    Grapdf := xbpGraPDF():New()
    GraPDF:Create()
    GraPDF:Page(XBPPRN_FORM_LETTER)

    GraPDF:Box( {0, 0}, {2110,2710} )

    GraPDF:Image( {10, 2500}, {2100, 2700}, "Logo.JPG", .f. )

    GraPDF:LineWidth( 4 )
    GraPDF:BoxColor( GRA_CLR_BLACK, GRA_CLR_PALEGRAY )
    GraPDF:Box( {150, 2300}, {1950, 2500} )
    GraPDF:TextColor( GRA_CLR_BLUE )
    GraPDF:font( '14.Arial Bold' )
    GraPDF:StringAt( {260, 2370}, "Parts Price List" )
    GraPDF:StringAt( {1700,2370}, dtoc(date()) )

    GraPDF:LineWidth( 1 )
    GraPDF:BoxColor( GRA_CLR_BLACK, GRA_CLR_WHITE )
    GraPDF:Box( {150, 700}, {1950, 2201} )
    GraPDF:Line( {450, 700}, {450, 2200} )
    GraPDF:Line( {1450, 700}, {1450, 2200} )
    GraPDF:Line( {1700, 700}, {1700, 2200} )

    GraPDF:TextColor( GRA_CLR_BLACK )
    GraPDF:font( '10.Courier New' )
    GraPDF:StringAt(  {160, 2100}, "Prt Nmbr" )
    GraPDF:StringAt(  {460, 2100}, "Descricao" )
    GraPDF:StringAt( {1460, 2100}, " Available" )
    GraPDF:StringAt( {1710, 2100}, "     Price" )
    GraPDF:Line( {150, 2095}, {1950, 2095} )

    go top
    line := 2040
    do while .not. eof()
        GraPDF:StringAt(  {160, line }, number )
        GraPDF:StringAt(  {460, line }, descrip )
```

```
        GraPDF:StringAt( {1460, line }, transform( qt, '###,###') )
        GraPDF:StringAt( {1710, line }, transform( price, '###,###.##') )
        line = line - 50
        skip
    enddo

    GraPDF:LineWidth( 3 )
    GraPDF:BoxColor( GRA_CLR_RED, GRA_CLR_PALEGRAY )
    GraPDF:Box( {150, 250}, {1950, 550} )
    GraPDF:TextColor( GRA_CLR_DARKBLUE )
    GraPDF:font( '20.Courier New Bold' )
    GraPDF:StringAt( {310, 450}, 'Demo Created by Softsupply Informatica' )
    GraPDF:TextColor( GRA_CLR_BLACK )
    GraPDF:font( '8.Courier New Italic' )
    GraPDF:StringAt( {200, 360}, 'Contact :' )
    GraPDF:StringAt( {200, 300}, 'eMail :' )
    GraPDF:StringAt( {1400, 360}, 'Fone    :' )
    GraPDF:StringAt( {1400, 300}, 'Fax   :' )
    GraPDF:font( '12.Courier New' )
    GraPDF:StringAt( {350, 360}, 'Edgar Borger' )
    GraPDF:StringAt( {1550, 360}, '(5511) 3159-1997' )
    GraPDF:StringAt( {1550, 300}, '(5511) 3255-5224' )
    GraPDF:TextColor( GRA_CLR_BLUE )
    GraPDF:StringAt( {350, 300}, 'eborger@terra.com.br' )

    GraPDF:Preview()

return
```
