2D

COLLABORATORS			
	TITLE : 2D		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 31, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

2D

1.1 2D Drawing library

```
Pure Basic - 2D Drawing library V1.00

  The 2D drawing is all the 2D rendering operations you can do
  on a visual area. Tracing a line, a box, a circle or even a text
  is condering as a 2D rendering operation. You just have to
  specify the output (screen, window, bitmap..) and draw...

Commands summary:

   BackColour
   BoxFill
   Circle
   Cls
   CopyBitmap
   CursX
   CursY
   DrawingFont
   DrawingMode
   DrawingOutput
   DrawingRastPort
   Ellipse
   FrontColour
   Locate
   Line
   ObtainBestPen
   Plot
   Point
   PrintText
   ReleasePen
   TextLength
   TextStyle

Example

   2D Drawing demo
```

1.2 plot

```
  SYNTAX
Plot(x,y)

  STATEMENT
Draw a plot in the active color (set by FrontColour()) in the current window.
```

1.3 boxfill

```
  SYNTAX
BoxFill(x1, y1, x2, y2)

  STATEMENT
Draw a filled box in the active color (set by FrontColour()) in the
current output.
```

1.4 backcolour

```
  SYNTAX
BackColour(Colour)

  STATEMENT
Set the default background colour for graphic functions and text display.
```

1.5 cursx

```
  SYNTAX
Result.w = CursX

  FUNCTION
Returns the text cursor X position in the current output.
```

1.6 cursy

```
  SYNTAX
Result.w = CursY

  FUNCTION
Returns the text cursor Y position in the current output.
```

1.7 frontcolour

```
  SYNTAX
FrontColour(Colour)

  STATEMENT
Sets the default foreground colour for graphic functions and text display.
```

1.8 locate

```
  SYNTAX
Locate(x,y)

  FUNCTION
Sets the text cursor to given position (for PrintText()).
```

1.9 printtext

```
  SYNTAX
PrintText(String$)

  STATEMENT
Display the given string  on  the  used  output  at  position  set  by
Locate() and with the colours set by FrontColour(). You can use TextStyle()
to change the output style (bold, italic, underline) and DrawingMode()
to change the way the text is rendered.
```

1.10 textstyle

```
  SYNTAX
TextStyle(Style)

  STATEMENT
Set the style for future text output.

The legal Style values are:

  0 = Nothing
  1 = Underline
  2 = Bold
  4 = Italic

You can mix these values if needed (Bold+Italic will be 2+4 = 6)
```

1.11 drawingfont

```
  SYNTAX
DrawingFont(FontID)

  STATEMENT
Sets the current font to given FontID. All new text rendered will
be done with the new specified font.

FontID MUST be a legal IntuiFont Pointer. You can use the NFont library
to load any font and pass the FontID() pointer to this function.
```

1.12 circle

```
  SYNTAX
Circle(x, y, Radius)

  STATEMENT
Draw an outlined circle at the position x,y with the size of Radius.
```

1.13 cls

```
  SYNTAX
Cls(Colour)

  STATEMENT
Set the entire output to the given colour.
```

1.14 copybitmap

```
  SYNTAX
CopyBitMap(BitMapID, SourceX, SourceY, DestX, DestY, Width, Height)

  STATEMENT
Copy the specified area situated in the given BitMapID, and determined
by SourceX, SourceY (x,y position in the bitmap to start the copy) and
Witdh, Height which give the size of the copied data. DestX and DestY
allow you to copy the data at this position in the current output.

This function is 100% OS Friendly and clipped.
```

1.15 drawingoutput

```
  SYNTAX
DrawingOutput(RastPort)

  STATEMENT
Set the drawing output to the specified rastport. After setting this, all
the Drawing commands are rendered on this rasport. You must specify a
```

```
valid rastport pointer.

  You can use the following functions to easily get rastports from objects:

    + WindowRastPort()
    + ScreenRastPort()
    + BitmapRastPort()

  Example:

  DrawingRastPort (WindowRastPort())  ; All drawing will be done on the window.
```

1.16 drawingmode

```
  SYNTAX
DrawingMode(Mode)

  STATEMENT
Change the drawing mode for graphics output:

  Here is a quick list of valid modes:

    #JAM1       = 0 : For text output, leave the background transparent.
    #JAM2       = 1 : For text output, print text with background colour
    #COMPLEMENT = 2 : XORred graphics
    #INVERSVID  = 4 : Use in conjunction with JAM2, it inverts background
                      and foreground colours when printing text.

Note: These constants are found in the AmigaOS resident file.
```

1.17 ellipse

```
  SYNTAX
Ellipse(x, y, RadiusX, RadiusY)

  STATEMENT
Draw an outlined ellipse at the position x,y with size of RadiusX
and RadiusY.
```

1.18 line

```
  SYNTAX
Line(x1, y1, x2, y2)

  STATEMENT
Draw a line in the active color (set by FrontColour()) on the current
output.
```

1.19 point

```
  SYNTAX
Colour.w = Point(x, y)

  STATEMENT
Return the colour number at the coordinates (x,y) in the current
output.
```

1.20 textlength

```
  SYNTAX
Length.w = TextLength(x, y)

  STATEMENT
Return the pixel length of the given string in the current output.
The main advantage of this function is so you can get the real length
of any strings used with any fonts (even non-proportional ones).
```

1.21 obtainbestpen

```
  SYNTAX
Result.w = ObtainBestPen(r, g, b, precision)

  STATEMENT
Return the colour number with the best match to the given parameter.
On a public screen with free colours, it will allocate a new colour
with the (r,g,b) value. The precision parameter tells the command
how it should be detected:

  Valid values of 'precision' (follow by system constante):
    Exact = -1 (#PRECISION_EXACT)
    Image =  0 (#PRECISION_IMAGE)
    Icon  = 16 (#PRECISION_ICON)
    Gui   = 32 (#PRECISION_GUI)

Note: You MUST use the ReleasePen(Result) for any colours you have allocated
before quitting your program, else the colours will never be released
to the free colour bank of the screen (very bad).
```

1.22 releasepen

```
  SYNTAX
ReleasePen(Pen)

  STATEMENT
Tell the system than this pen isn't used (locked) by your program, so
that other programs can use it. You must call this function if you have
allocated any pens with ObtainBestPen().
```

1.23 drawingrastport

```
  SYNTAX
*RastPort = DrawingRastPort()

  STATEMENT
Return the value of the currently used RastPort for output (for
advanced programmers).
```