

RESUMO – Principais Comandos gráficos do FreeBASIC

SCREEN – Seleciona um modo de tela gráfica (veja os modos de tela disponíveis na ajuda)

SCREENRES – Seleciona uma tela gráfica com uma dada resolução (veja na ajuda)

PSET e **PRESET** – Desenha e Apaga um ponto na tela

LINE – Desenha linhas ou retângulos na tela gráfica

CIRCLE – Desenha circunferências, elipses ou arcos na tela gráfica

DRAW – Desenha na tela gráfica (é um cursor "livre" na tela)

DRAW STRING – Desenha strings na tela gráfica

COLOR – Especifica a cor de frente e de fundo da tela gráfica

PAINT – Pinta com a cor especificada no último comando COLOR

GET MOUSE – Leitura das coordenadas (x,y) do cursor e do estado dos botões do mouse

SET MOUSE – Posicionamento do cursor do mouse nas coordenadas (x,y) da tela gráfica

PROCEDIMENTOS GRÁFICOS A SEREM IMPLEMENTADOS:

Conversão de coordenadas **World** (mundo real) \Longleftrightarrow **View** (tela gráfica):

É necessário fazer a conversão de coordenadas para se preservar as proporções métricas do gráfico que está sendo impresso na tela gráfica do computador com o gráfico do mundo real, desejado pelo usuário

Coordenadas **World**: são as coordenadas do mundo real do usuário

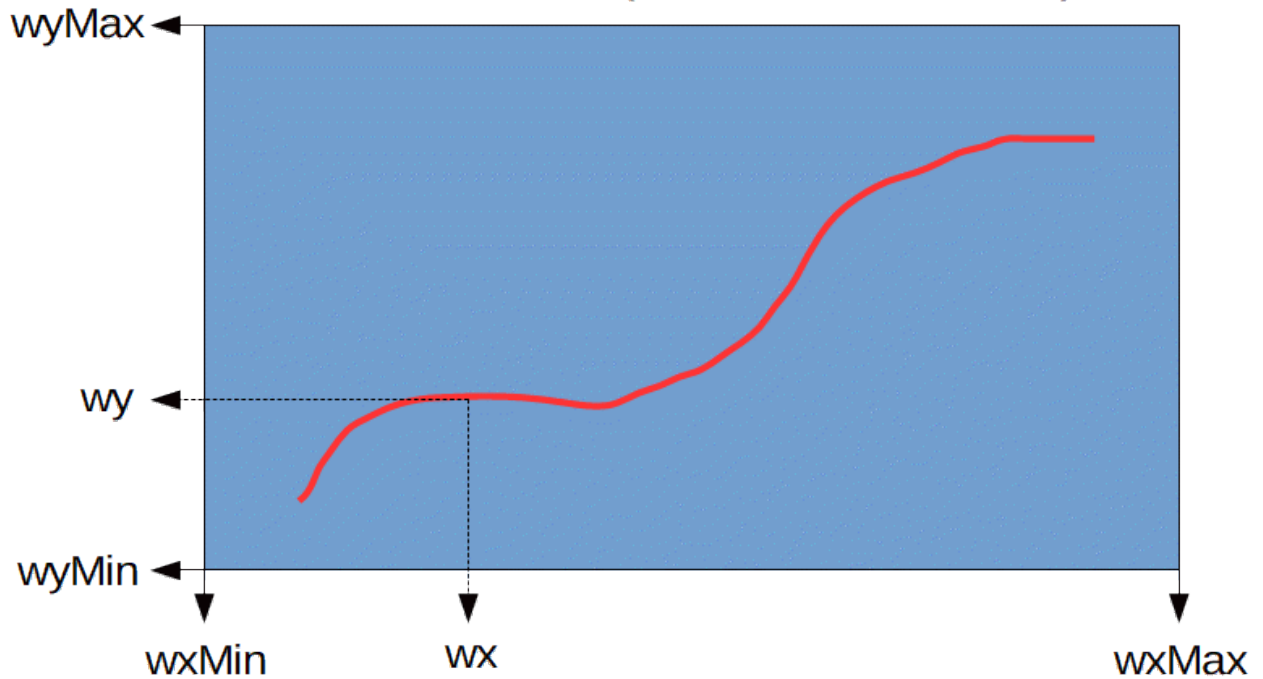
Coordenadas **View**: são coordenadas da tela gráfica do computador (depende da resolução da tela gráfica, em pixels)

A Figura da página seguinte mostra as coordenadas gráficas mais importantes para se realizar estas conversões, sendo que para se ter proporcionalidade entre os gráficos do mundo real e da tela gráfica, são necessárias que as seguintes razões e proporções sejam obedecidas:

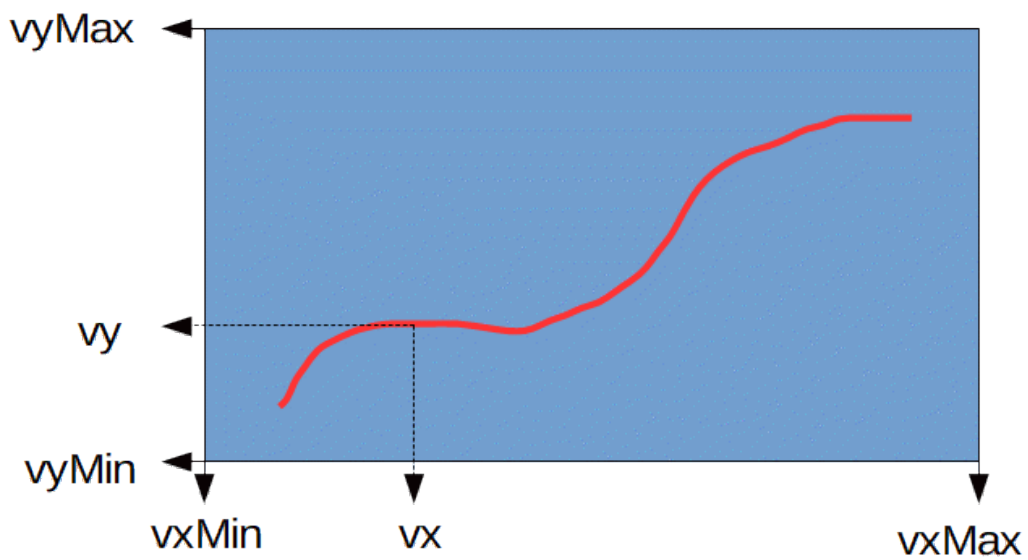
$$\frac{(wx - wxMin)}{(wxMax - wxMin)} = \frac{(vx - vxMin)}{(vxMax - vxMin)} \quad \text{Equação 1 (Para coordenada x)}$$

$$\frac{(wy - wyMin)}{(wyMax - wyMin)} = \frac{(vy - vyMin)}{(vyMax - vyMin)} \quad \text{Equação 2 (Para coordenada y)}$$

Coordenadas World (mundo real do usuário)



Coordenadas View (tela gráfica do computador)



Pode-se rearranjar as Equações 1 e 2 para se obter as seguintes funções:

$$vx = (wx - wxMin) * \frac{(vxMax - vxMin)}{(wxMax - wxMin)} + vxMin \quad \text{Função 1 (wxTOvx)}$$

$$vy = (wy - wyMin) * \frac{(vyMax - vyMin)}{(wyMax - wyMin)} + vyMin \quad \text{Função 2 (wyTOvy)}$$

$$wx = (vx - vxMin) * \frac{(wxMax - wxMin)}{(vxMax - vxMin)} + wxMin \quad \text{Função 3 (vxTOWx)}$$

$$wy = (vy - vyMin) * \frac{(wyMax - wyMin)}{(vyMax - vyMin)} + wyMin \quad \text{Função 4 (vyTOWy)}$$

As funções acima podem ser **otimizadas** calculando-se os fatores **fatorWxVx** e **fatorWyVy** que relacionam a razão das coordenadas limites do gráficos world e view:

$$fatorWxVx = \frac{(vxMax - vxMin)}{(wxMax - wxMin)} \quad fatorWyVy = \frac{(vyMax - vyMin)}{(wyMax - wyMin)}$$

$$vx = (wx - wxMin) * fatorWxVx + vxMin \quad \text{Função 1 (wxTOvx)}$$

$$vy = (wy - wyMin) * fatorWyVy + vyMin \quad \text{Função 2 (wyTOvy)}$$

$$wx = (vx - vxMin) * \frac{1}{fatorWxVx} + wxMin \quad \text{Função 3 (vxTOWx)}$$

$$wy = (vy - vyMin) * \frac{1}{fatorWyVy} + wyMin \quad \text{Função 4 (vyTOWy)}$$

Portanto, uma biblioteca de procedimentos gráficos em BASIC deve inicialmente acessar um procedimento de inicialização da tela gráfica para obter os valores limites das coordenadas minimas e máximas dos mundos do usuário (World) e da tela gráfica (View), para depois calcular os fatores (**fatorWxVx** e **fatorWyVy**) armazenados em variaveis globais. Adicionalmente, est biblioteca deve conter as funções (FUNCTIONS) de conversão de coordenadas propriamente ditas, conforme mostrado abaixo:

Uma biblioteca gráfica poderia ser implementada com estes exemplos de declarações e procedimentos abaixo:

'Declaração de variáveis na biblioteca gráfica (Graficos.bas ou Graficos.bi), a ser declarada antes dos procedimentos:

,

'UDT: "User Defined Type" ou "Tipo Definido pelo Usuario" para encapsular variáveis de diferentes tipos

```
TYPE limitesDoGrafico
    'Coords world
    DIM wxMin AS DOUBLE
    DIM wxMax AS DOUBLE
    DIM wyMin AS DOUBLE
    DIM wyMax AS DOUBLE
    'Coords view
    DIM vxMin AS INTEGER
    DIM vxMax AS INTEGER
    DIM vyMin AS INTEGER
    DIM vyMax AS INTEGER
END TYPE
```

'A variável **global** lg encapsula as variáveis limites do gráfico acima:

```
DIM SHARED lg AS limitesDoGrafico
```

,

'Fatores para inicializacao da tela (variaveis **globais**)

```
DIM SHARED fatorWxVx AS DOUBLE
DIM SHARED fatorWyVy AS DOUBLE
```

'-----

'-----

'As seguintes **FUNCTION**s devem ser implementadas, para se converter coordenadas
'world ↔ view: wx ↔ vx e wy ↔ vy:

'-----

FUNCTION wxTOvx (**BYREF** wx **AS DOUBLE**) **AS INTEGER**

'Converte coordenada wx em coordenada vx

WITH lg

DIM vx **AS INTEGER**

vx = **CINT**((wx - .wxMin) * fatorWxVx + .vxMin)

RETURN vx

END WITH

END FUNCTION

'-----

FUNCTION wyTOvy (**BYREF** wy **AS DOUBLE**) **AS INTEGER**

'Converte coordenada wy em coordenada vy

WITH lg

DIM vy **AS INTEGER**

vy = **CINT**((wy - .wyMin) * fatorWyVy + .vyMin)

RETURN vy

END WITH

END FUNCTION

'-----

FUNCTION vxTOwx (**BYREF** vx **AS INTEGER**) **AS DOUBLE**

'Converte coordenada vx em coordenada wx

WITH lg

DIM wx **AS DOUBLE**

wx = (vx - .vxMin) / fatorWxVx + .wxMin

RETURN wx

END WITH

END FUNCTION

'-----

FUNCTION vyTOwy (**BYREF** vy **AS INTEGER**) **AS DOUBLE**

'Converte coordenada vy em coordenada wy

WITH lg

DIM wy **AS DOUBLE**

wy = (vy - .vyMin) / fatorWyVy + .wyMin

RETURN wy

END WITH

END FUNCTION

'-----

Os seguintes procedimentos são necessários para inicializações de variáveis e da Tela Gráfica, e para a plotagem dos pontos:

```
'-----  
SUB InicializaCoordsView  
    'Inicializa as variáveis inteiras que definem os limites do gráfico (em pixels)  
    DIM margem AS INTEGER = 50  
    ' Tela grafica com resolucao 640x480, 256 colors (8bits)  
    SCREENRES (640,480,8)  
    WITH lg  
        .VxMin = margem  
        .VxMax = 639-margem  
        .VyMin = 479-margem  
        .VyMax = margem  
    END WITH  
    'OBS: VyMin e VyMax definidos desta maneira, garantem que o eixo y  
    'do grafico ira ser plotado em ordem crescente na direcao correta, ou seja, para cima  
END SUB  
  
'-----  
  
SUB InicializaCoordsWorld (BYVAL n AS INTEGER, x() AS DOUBLE, y() AS DOUBLE)  
    'Inicializa as coordenadas World do usuário em variáveis reais ( precisão double)  
    WITH lg  
        .WxMin = x(1)  
        .WxMax = x(1)  
        .WyMin = y(1)  
        .WyMax = y(1)  
        FOR i AS INTEGER = 2 TO n  
            IF x(i) < .WxMin THEN .WxMin = x(i)  
            IF x(i) > .WxMax THEN .WxMax = x(i)  
            IF y(i) < .WyMin THEN .WyMin = y(i)  
            IF y(i) > .WyMax THEN .WyMax = y(i)  
        NEXT  
    END WITH  
END SUB  
  
'-----  
  
SUB InicializaFatores  
    'Este procedimento deve ser chamado  
    'antes das FUNCTIONS de conversao de coordenadas world <---> view,  
    'ou seja, antes da plotagem dos pontos  
    WITH lg  
        fatorWxVx = (.vxMax - .vxMin) / (.wxMax - .wxMin)  
        fatorWyVy = (.vyMax - .vyMin) / (.wyMax - .wyMin)  
    END WITH  
END SUB
```

'-----
'Este procedimento PlotarPontos é o único procedimento que deve ser público, ou seja, a ser
'chamado pelo programa principal pois ele já executa automaticamente os procedimentos de
'inicializacao de variáveis e da Tela gráfica:

SUB PlotarPontos (**BYVAL** n **AS INTEGER**, x() **AS DOUBLE**, y() **AS DOUBLE**)

'Plota os n pontos com coordenadas armazenadas nos vetores x() e y()

DIM raio **AS INTEGER** = 5 'numero minimo de pixeis para boa definicao do circulo

DIM cor **AS INTEGER** = 2 'cor verde

DIM **AS INTEGER** Vx1, Vx2

DIM **AS INTEGER** Vy1, Vy2

'Os seguintes procedimentos de inicializacao devem ser chamados
'antes da plotagem dos pontos, **nesta ordem**:

InicializaCoordsView

InicializaCoordsWorld (n, x(), y())

InicializaFatores

'Faz a moldura (limites das coords view) na tela grafica:

LINE (lg.vxmin,lg.vymin)-(lg.vxmax,lg.vymax),15,B

'Plota o primeiro ponto

Vx1 = WxToVx (x(1))

Vy1 = WyToVy (y(1))

CIRCLE (Vx1,Vy1), raio, cor

'Plota os demais pontos, ligando-os por um segmento de reta:

FOR i **AS INTEGER** = 2 **TO** n

'Segundo ponto

Vx2 = WxToVx (x(i))

Vy2 = WyToVy (y(i))

LINE (Vx1,Vy1) - (Vx2,Vy2), cor

CIRCLE (Vx2,Vy2), raio, cor

'Armazena o segundo ponto em variaveis1

Vx1 = Vx2

Vy1 = Vy2

NEXT

END SUB