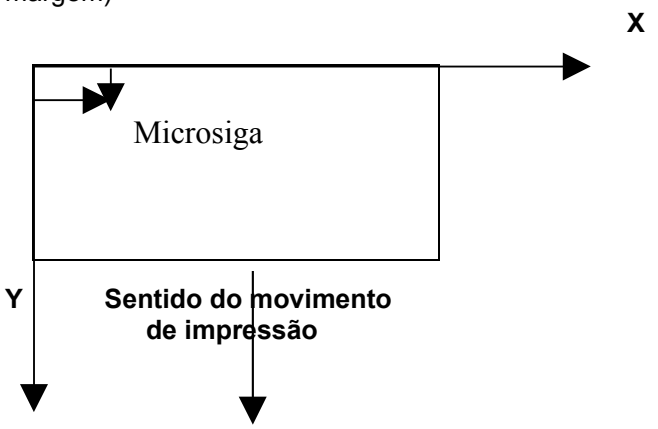
GUIA DE REFERÊNCIA RÁPIDA

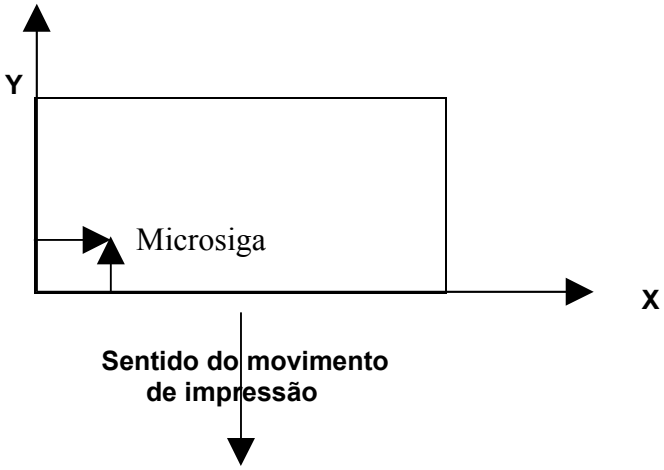
ÍNDICE

Funções para impressão de etiquetas padrão ZPL,ALLEGRO e ELTRON

Visualização do posicionamento da imagem na etiqueta no padrão Zebra (ZPL,EPL):
(OBS: Alguns modelos de impressoras ELTRON, possuem o alinhamento da folha de etiqueta centralizado, por isso deve-se considerar o ponto de impressão da posição+a margem)



Visualização do posicionamento da imagem na etiqueta no padrão Allegro:



Nota:
Parâmetros que estiverem entre [], significa que não são OBRIGATÓRIOS, os parâmetros que estiverem com (*), significa que são ignorados no padrão Allegro e Eltron.

MSCBPrinter

Tipo: Impressão

Configura modelo da impressora, saída utilizada, resolução na impressão e tamanho da etiqueta a ser impresso.

Parâmetros

[ModelPrt] = String com o modelo de impressora:
Zebra: S400, S600, S500-6, Z105S-6, Z16S-6, S300, S500-8, Z105S-8, Z160S-8, Z140XI, Z90XI e Z170ZI.
Allegro:
ALLEGRO, PRODIGY, DMX e DESTINY.

Eltron:
ELTRON E ARGOX
cPorta = String com a porta
[nDensidade] = Numero com a densidade referente a quantidade de pixel por mm
[nTamanho] = Tamanho da etiqueta em Milímetros.
[lSrv] = Se .t. imprime no server,.f. no client
[nPorta] = numero da porta de outro server
[cServer] = endereço IP de outro server
[cEnv] = environment do outro server
[nMemoria] = Numero com bloco de memória

Observações:
O parâmetro nDensidade não é necessário informar, pois ModelPrt o atualizará automaticamente. A utilização deste parâmetro (nDensidade) deverá ser quando não souber o modelo da impressora, a aplicação entendera que se trata de uma impressora Zebra.
O tamanho da etiqueta será necessário quando a mesma não for continua.

Exemplo
MSCBPRINTER("S500-8", "COM2:9600,e,7,2",NIL, 42)
MSCBPRINTER("S500-8", "LPT1",NIL, 42)
MSCBPRINTER("S600","COM1:9600,N,8,2",NIL,NIL,.T.,1024,"SERVER-AP","ENVCODEBASEPORT609")
MSCBPRINTER("S600", "LPT1",NIL, 42,.F.,NIL,NIL,NIL,10240)

MSCBClosePrinter
Tipo: Impressão

Finaliza a conexão com a impressora

Exemplo
MSCBClosePrinter()

MSCBBegin
Tipo: Impressão
Inicializa a montagem da imagem para cada etiqueta

Parâmetros
[nQtde] = Quantidade de cópias
[nVeloc] = Velocidade (1,2,3,4,5,6) polegadas por segundo
[nTamanho]= Tamanho da etiqueta em Milímetros.

Exemplo:
MSCBBEGIN(3,4,40)

MSCBEnd
Tipo: Impressão
Finaliza a montagem da imagem

Exemplo
MSCBEND()

MSCBSay
Tipo: Impressão
Imprime uma String

Parâmetros

nXmm = Posição X em Milímetros
nYmm = Posição Y em Milímetros
cTexto = String a ser impressa
cRotação = String com o tipo de Rotação (N,R,I,B)
N-Normal
R-Cima p/baixo
I-Invertido
B-Baixo p/ Cima
cFonte = String com os tipos de Fonte
Zebra: (A,B,C,D,E,F,G,H,0) 0(zero)- fonte escalar
Allegro: (0,1,2,3,4,5,6,7,8,9) 9 – fonte escalar
Eltron: (0,1,2,3,4,5)
cTam = String com o tamanho da Fonte
*[lReverso]= Imprime em reverso quando tiver sobre um box preto
[lSerial] = Serializa o código
[cIncr] = Incrementa quando for serial positivo ou negativo
*[lZerosL] = Coloca zeros a esquerda no numero serial

Exemplo

MSCBSAY(15,3,"MICROSIGA ","N","C","018,010",.t.)
MSCBSAY(15,3,"MICROSIGA ","N","C","018,010",.t.,.t.,"3",.t.)

MSCBSayBar

Tipo: Impressão

Imprime Código de Barras

Parâmetros

nXmm = Posição X em Milímetros
nYmm = Posição Y em Milímetros
cConteudo = String a ser impressa
cRotação = String com o tipo de Rotação
cTypePrt = String com o Modelo de Código de Barras
Zebra:
2 - Interleaved 2 of 5
3 - Code 39
8 - EAN 8
E - EAN 13
U - UPC A
9 - UPC E
C - CODE 128
Allegro:
D - Interleaved 2 of 5
A - Code 39
G - EAN 8
F - EAN 13
B - UPC A
C - UPC E
E - CODE 128
Eltron:
2 - Interleaved 2 of 5
3 - Code 39
E80 - EAN 8
E30 - EAN 13
UA0 - UPC A
UE0 - UPC E
1 - CODE 128

[nAltura] = Altura do código de Barras em Milímetros
*[lDigver] = Imprime dígito de verificação
[lLinha] = Imprime a linha de código
*[lLinBaixo]= Imprime a linha de código acima das barras

[cSubSetIni]= Utilizado no code128
[nLargura] = Largura da barra mais fina em pontos default 3
[nRelacao] = Relação entre as barras finas e grossas em pontos default 2
[lCompacta] = Compacta o código de barra
[lSerial] = Serializa o código
[cIncr] = Incrementa quando for serial positivo ou negativo
*[lZerosL] = Coloca zeros a esquerda no numero serial

Exemplo

MSCBSAYBAR(20,22,AllTrim(SB1->B1_CODBAR),"N","C",13)

***MSCBSayMemo**

Tipo: Impressão

Monta e imprime um campo MEMO

Parâmetros

nXmm = Posição X em Milímetros
nYmm = Posição Y em Milímetros
nLMemomm = Tamanho da 1 linha do campo memo em Milímetros
nQLinhas = Quantidade de linhas
cTexto = String a ser impressa
cRotação = String com o tipo de Rotação (N,R,I,B)
cFonte = String com o tipo de Fonte (A,B,C,D,E,F,G,H,0)
cTam = String com o tamanho da Fonte
[lReverso]= lógica se imprime em reverso quando tiver
sobre um box preto
[cAlign] = String com o tipo de Alinhamento do memo
L - Esquerda
R - Direita
C - Centro
J - Margem a margem (justificado)

Exemplo

MSCBSAYMEMO (1,10,8,4,cTexto,"N","A","9,5",.f.,"C")

MSCBBox

Tipo: Impressão

Imprime um Box

Parâmetros

nX1mm = Posição X1 em Milímetros
nY1mm = Posição Y1 em Milímetros
nX2mm = Posição X2 em Milímetros
nY2mm = Posição Y2 em Milímetros
[nEspessura]= Numero com a espessura em pixel
*[cCor] = String com a Cor Branca ou Preta ("W" ou "B")

Exemplo

MSCBBOX(12,01,31,10,37)

MSCBLineH

Tipo: Impressão

Imprime uma linha horizontal

Parâmetros

nX1mm = Posição X1 em Milímetros
nY1mm = Posição Y1 em Milímetros
nX2mm = Posição X2 em Milímetros
[nEspessura]= Numero com a espessura em pixel
*[cCor] = String com a Cor Branca ou Preta ("W" ou "B")

Exemplo

MSCBLineH(01,10,80,3,"B")

MSCBLineV

Tipo: Impressão

Imprime uma linha vertical

Parâmetros

nX1mm = Posição X1 em Milímetros
nY1mm = Posição Y1 em Milímetros
nY2mm = Posição X2 em Milímetros
[nEspessura] = Numero com a espessura em pixel
*[cCor] = String com a Cor Branca ou Preta ("W" ou "B")

Exemplo

MSCBLineV(01,10,80,3,"B")

MSCBLoadGrf

Tipo: Impressão

Carrega uma imagem para memória da impressora

Observações

Para o padrão Zebra, arquivo do gráfico tem que ser do tipo GRF, gerado através de um PCX ou TIF no software fornecido pelo fabricante da zebra.

Para o padrão Allegro e Eltron, arquivo do gráfico pode ser do tipo BMP, PCX e IMG. Não precisa ser convertido.

Parâmetros

cImagem = nome do arquivo que será carregado

Exemplo

MSCBLOADGRF("LOGO.GRF")

MSCBGrafic

Tipo: Impressão

Imprime gráfico que está armazenado na memória da impressora

Parâmetros

nXmm = Posição X em Milímetros
nYmm = Posição Y em Milímetros
cArquivo = Nome do gráfico que foi carregado na memória da impressora (não colocar a extensão .GRF)
*[lReverso]= Imprime em reverso quando tiver sobre um box preto

Exemplo

MSCBGRAFIC(3,3,"LOGO",.t.)

MSCBChkStatus

Tipo: Impressão

Seta ou visualiza o controle de status do sistema com a impressora.

Parâmetros

[lStatus] = Lógica ativa/desativa o controle

Retorno

Retorna o Status

Com o status ativado, sempre que a aplicação enviar qualquer informação para a impressora, será analisado o status, caso esteja com o buffer cheio, sem papel ou sem ribbon, o sistema aguardara até que os itens anteriores estejam solucionados.

Exemplo

MSCBCHKSTATUS(.f.) // desativa a verificação

MSCBWrite

Tipo: Impressão

Permite enviar para porta uma linha de programação nativa da Impressora.

Parâmetros

cConteudo = Linha de programação nativa da impressora.

Exemplo

MSCBWRITE("^FO1,1^GB400,50,25^FS")

Tipos de fontes para Zebra:

As fontes A,B,C,D,E,F,G,H, são do tipo BITMAPPEDs, tem tamanhos definidos e podem ser expandidas proporcionalmente as dimensões mínimas.

Exemplo: Fonte do tipo A, 9 X 5 ou 18 x 10 ou 27 X 15 …

A fonte 0 (Zero) é do tipo ESCALAR, esta será gerada na memória da impressora, portanto torna-se um processamento lento.

Tipos de fontes para Allegro:

As fontes 0,1,2,3,4,5,6,7,8, são do tipo BITMAPPEDs, tem tamanho definido e podem ser expandidos.

Exemplo: Fonte do tipo 1, 1X1 ou 1x2 ou 1x3 ou 2x1 ou 2x2…

A fonte 9 (Nove) é do tipo ESCALAR, esta será gerada na memória da impressora, portanto torna-se um processamento lento, as dimensões deste tipo de fonte tem que ser passando junto com o tamanho da fonte.

Exemplo. cTam := “001,001,002”

```
                \_____/ \_/
                   |    |
                   |    |→ dimensão da fonte que pode ser de 0 à 9
                   |→ tamanho da fonte
```

Exemplo padrão Zebra.

```
User Function ETI_ZEBRA()
Local nX
Local cPorta
cPorta := "COM2:9600,n,8,2"
MSCBPRINTER("S500-8",cPorta,,,.f.,,,,)
MSCBLOADGRF("SIGA.GRF")
For nx:=1 to 3
        MSCBBEGIN(1,6)
        MSCBBOX(02,01,76,35)
        MSCBLineH(30,05,76,3)
        MSCBLineH(02,13,76,3,"B")
        MSCBLineH(02,20,76,3,"B")
        MSCBLineV(30,01,13)
        MSCBGRAFIC(2,3,"SIGA")
        MSCBSAY(33,02,'PRODUTO',"N","0","025,035")
        MSCBSAY(33,06,"CODIGO","N","A","015,008")
        MSCBSAY(33,09, "000006", "N", "0", "032,035")
        MSCBSAY(05,14,"DESCRICAO","N","A","015,008")
        MSCBSAY(05,17,"IMPRESSORA ZEBRA S500-8","N", "0", "020,030")
        MSCBSAYBAR(23,22,"00000006","N","C",8.36,.F.,.T.,.F.,,2,1,.F.,.F.,"1",.T.)
        MSCBEND()
Next
```

```
MSCBCLOSEPRINTER()
Return
```

Exemplo padrão Allegro.

```
User Function ETI_ALLEGRO()
Local nX
Local cPorta
cPorta := "COM2:9600,n,8,2"
MSCBPRINTER("ALLEGRO",cPorta,,,.f.,,,,)
MSCBLOADGRF("SIGA.BMP")
For nx:=1 to 3
        MSCBBEGIN(1,6)
        MSCBBOX(02,01,76,34,1)
        MSCBLineH(30,30,76,1)
        MSCBLineH(02,23,76,1)
        MSCBLineH(02,15,76,1)
        MSCBLineV(30,23,34,1)
        MSCBGRAFIC(2,26,"SIGA")
        MSCBSAY(33,31,'PRODUTO',"N","2","01,01")
        MSCBSAY(33,27,"CODIGO","N","2","01,01")
        MSCBSAY(33,24, "000006", "N", "2", "01,01")
        MSCBSAY(05,20,"DESCRICAO","N","2","01,01")
        MSCBSAY(05,16,"IMPRESSORA ALLEGRO 2 BR","N", "2", "01,01")
        MSCBSAYBAR(22,03,"00000006","N","E",8.36,.F.,.T.,.F.,,3,2,.F.,.F.,"1",.T.)
        MSCBEND()
Next
MSCBCLOSEPRINTER()
Return
```

Exemplo padrão Eltron.

```
User Function ETI_ELTRON()
Local nX
Local cPorta
cPorta := "COM2:9600,n,8,2"
MSCBPRINTER("ELTRON",cPorta,,,.f.,,,,)
MSCBCHKStatus(.f.)
MSCBLOADGRF("SIGA.PCX")
For nx:=1 to 3
        MSCBBEGIN(1,6)
        MSCBGRAFIC(04,02,"SIGA")
        MSCBBOX(05,01,76,30,2)
        MSCBLineH(30,06,71,2)
        MSCBLineH(05,12,71,2)
        MSCBLineH(05,18,71,2)
        MSCBLineV(30,1,1.3,90) //Monta Linha Vertical
        MSCBSAY(33,02,'PRODUTO',"N","2","1,2")
        MSCBSAY(33,07,"CODIGO", "N", "1", "1,1")
        MSCBSAY(33,09,"000006", "N", "1", "1,2")
        MSCBSAY(07,13,"DESCRICAO","N","1","1,1")
        MSCBSAY(07,15,"IMPRESSORA ELTRON TLP2742","N", "1", "1,2")
        MSCBSAYBAR(28,19,"00000006",'N','1',50,.T.,,,,2,2,,,,)
        MSCBEND()
Next
MSCBCLOSEPRINTER()
Return
```

COMANDOS PARA TELNET VT100.

@...VTSay

Tipo: TELNET VT100

Exibe dados em uma linha e coluna especificadas

Sintaxe

@ <nLin>, <nCol>
[VTSAY <exp> [PICTURE <cSayPicture>]]

Parâmetros

<nLin> e <nCol> são as coordenadas de linha e coluna da sa¡da.
Os valores de linha podem variar entre zero e VTMAXROW(). O mesmo vale para as colunas.
VTSAY <exp> exibe o resultado de uma expressão de qualquer tipo.
PICTURE <cSayPicture> define a mascara para a saída de exp.

Exemplo

```
nQtd    :=15.45
cDesc  := "Teste descricao"
//
@ 1, 1 VTSAY nQtd PICTURE "@9999.99"
@ 2, 1 VTSAY “Teste VTG100”
@ 4, 1 VTSAY cDesc PICTURE "@!"
```

VTClear Screen

Tipo: TELNET VT100

Apaga a tela e coloca o cursor na posição inicial

Sintaxe

VTCLEAR [SCREEN]

@...VTGet

Tipo: TELNET VT100

Cria um novo objeto VTGET e o coloca em exibição na tela

Sintaxe

```
@ <nLin>, <nCol>
  [VTSAY <exp>
    [PICTURE <cSayPicture>]]
  VTGET <idVar>
    [PICTURE <cGetPicture>]
    [WHEN <lPreCondicao>]
    [VALID <lPosCondicao>]
    [PASSWORD]
    [F3<tabela>]
```

Parâmetros

<nLin> e <nCol> São as coordenadas de linha e coluna para a operação. Se a clausula VTSAY está presente, especificam as coordenadas para o VTSAY, e o VTGET , exibido a direita deste.

VTSAY exibe o valor de <exp> nas coordenadas especificadas. Caso a PICTURE <cSayPicture> seja especificada.

VTGET <idVar> define o nome da variável de qualquer tipo de dados a ser editada. Ela pode ser caractere, data ou numérica .

PICTURE <cGetPicture> especifica uma mascara para exibição e as regras para edição do VTGET.

WHEN <lPreCondicao> especifica uma expressão que deve ser satisfeita antes do cursor entrar na região de edição de VTGET. Se <lCondicao> , avaliada como verdadeira (.T.), , permitido ao cursor entrar; de outra forma, o VTGET corrente , saltado e o cursor move-se para o próximo VTGET.

VALID <lPosCondicao> especifica uma expressão que deve ser satisfeita antes que o cursor possa deixar a região de edição do VTGET corrente. O VALID<lPosCondicao> , avaliado sempre que o usuário tenta deixar a região de edição do VTGET, a menos que a tecla Esc seja pressionada . Se <lPosCondicao> retorna falso (.F.), o controle retorna ao VTGET e o usuário não pode deixa -lo até que <lPosCondicao> retorne verdadeiro (.T.) ou o usuário aperte Esc. Um VALID <lPosCondicao> pode conter ou ser uma função definida pelo usuário, permitindo-lhe executar buscas e outros tipos de operações de validação.

PASSWORD Monta o VTGET para entrada de dados com * na tela, utilizado para SENHAS

F3 Associa este VTGET a uma tabela do SXB ou Sx5.

Descrição

Quando um comando VTREAD , especificado, um VTGET executa uma edição do conteúdo de <idVar> de qualquer tipo de dado. Quando um objeto VTGET , criado, o nome e valor corrente de <idVar> são guardados no objeto VTGET. O valor de <idVar> fica armazenado no que, chamado de buffer do VTGET. O buffer de VTGET , o que , realmente mostrado na tela e editado.

Exemplo

```
#include 'apvt100.ch'
 nNumber = 0
 @ 0, 0 VTSAY "Digite um numero";
   VTGET nNumber;
   VALID nNumber > 0
```

VTRead

Tipo: TELNET VT100

Ativa edição em tela usando objetos GET

Sintaxe

```
VTREAD
```

Descrição

O comando READ executa um módulo de edição em tela usando todos os objetos VTGET criados e adicionados.

Dentro de um READ, o usuário pode editar o buffer de cada objeto VTGET bem como mover-se de um objeto GET para outro. Antes que o usuário possa entrar com um objeto VTGET, o controle passa para o respectivo WHEN.

Quando o usuário pressiona uma tecla de saída de VTGET, o controle passa VALID respectivo, caso tenha sido especificada.

O exemplo abaixo define vários VTGETs e a seguir usa o comando READ:

```
#include 'apvt100.ch'
  cVar1 := cVar2 := cVar3 := SPACE(10)
  @ 1, 1 VTSAY "Um  :" VTGET cVar1 VALID !EMPTY(cVar1)
  @ 2, 1 VTSAY "Dois:" VTGET cVar2 WHEN RTRIM(cVar1) != "Um"
  @ 3, 1 VTSAY "Tres:" VTGET cVar3 VALID !EMPTY(cVar3)
  VTREAD
```

VTSave Screen

Tipo: TELNET VT100

Grava a tela corrente numa variável

Sintaxe

VTSAVE SCREEN TO <idVar>

Parâmetros

TO <idVar> especifica a variável que serão atribuídos os conteúdos da tela corrente.

Exemplo

```
#include 'apvt100.ch'
VTSave Screen To aTela
cVar1 := cVar2 := cVar3 := SPACE(10)
@ 1, 1 VTSAY " Um  :" VTGET cVar1 VALID !EMPTY(cVar1)
@ 2, 1 VTSAY " Dois:" VTGET cVar2 WHEN RTRIM(cVar1) != "Um"
@ 3, 1 VTSAY " Tres:" VTGET cVar3 VALID !EMPTY(cVar3)
VTREAD
VTRestore Screen From aTela
```

VTRestore Screen

Tipo: TELNET VT100

Exibe uma tela guardada

Sintaxe

VTRESTORE SCREEN [FROM <aTela>]

Parâmetros

FROM <aTela> especifica uma variável que contem o conteúdo da tela a ser exibida.

Exemplo

```
#include 'apvt100.ch'
VTSave Screen To aTela
cVar1 := cVar2 := cVar3 := SPACE(10)
@ 1, 1 VTSAY " Um  :" VTGET cVar1 VALID !EMPTY(cVar1)
@ 2, 1 VTSAY " Dois:" VTGET cVar2 WHEN RTRIM(cVar1) != "Um"
@ 3, 1 VTSAY " Tres:" VTGET cVar3 VALID !EMPTY(cVar3)
```

VTREAD
VTRestore Screen From aTela

VTPause

Tipo: TELNET VT100

Suspende a execução de um programa até que seja pressionada a tecla ENTER

Sintaxe

#include 'apvt100.ch'
VTPAUSE

@...VTPause

Tipo: TELNET VT100

Exibe dados em uma linha e coluna especificadas e para a execução de um programa até que seja pressionada a tecla ENTER

Sintaxe

@ <nLin>, <nCol>
[VTPAUSE <exp> [PICTURE <cSayPicture>]]

Parâmetros

<nLin> e <nCol> são as coordenadas de linha e coluna da sa¡da.
Os valores de linha podem variar entre zero e VTMAXROW(). O mesmo vale para as colunas.
VTPAUSE <exp> exibe o resultado de uma expressão de qualquer tipo.
PICTURE <cSayPicture> define a mascara para a saída de exp.

Exemplo

#include 'apvt100.ch'
@ 7, 1 VTPause "Tecle ENTER p/ Continuar"

VTSetSize

Tipo: TELNET VT100

Seta o limite da área de trabalho.

Sintaxe

VTSETSIZE <nLin>, <nCol>

Parâmetros

<nLin> e <nCol> são as coordenadas máximas de linha e coluna.

Exemplo

#include 'apvt100.ch'
VTSetSize 8,20

VTSet Key

Tipo: TELNET VT100

Atribui a chamada de uma rotina a uma tecla

Sintaxe

VTSET KEY <nCodigoTecla> TO [<idRotina>]

Parâmetros

<nCodigoTecla> , o valor VTINKEY() da tecla a qual se atribui a rotina.
TO <idRotina> especifica o nome da rotina que é executada quando se aperta uma tecla.

Se <idRotina> não é especificada, a definição corrente é liberada.

Exemplo

Este exemplo demonstra como usar VTSET KEY para invocar uma rotina quando o usuário aperta. A tecla 'A'.

```
#include 'apvt100.ch'
VTSET KEY 65 TO TESTE
CCodigo := space(6)
@ 1, 1 VTGET cCodigo
VTREAD
RETURN

FUNCTION Teste()
  @ 2,1 VTSay 'TESTE'
RETURN NIL
```

FUNCOES PARA TELNET VT100

VTReadVar()

Tipo: TELNET VT100

Retorna o nome da variável VTGET corrente

Sintaxe

VTREADVAR() --> cNomeVar

Retorna

VTREADVAR() retorna o nome da variável associada ao objeto VTGET corrente.

VTSave()

Tipo: TELNET VT100

Grava uma região de tela para posterior exibição

Sintaxe

VTSAVE (<nTopo>, <nEsquerda>, <nBase>, <nDireita>) --> cTela

Parâmetros

<nTopo>, <nEsquerda>, <nBase>, e <nDireita> definem as coordenadas da região de tela a ser gravada. Caso <nBase> ou <nDireita> seja maior do que VTMAXROW() ou VTMAXCOL(), a tela é cortada.

Retorna

VTSAVE () retorna a região de tela especificada na forma de uma cadeia de caracteres.

Exemplo

```
#include 'apvt100.ch'
aTela := VTSave(0,0,4,10)
cVar1 := cVar2 := cVar3 := SPACE(10)
@ 1, 1 VTSAY " Um  :" VTGET cVar1 VALID !EMPTY(cVar1)
@ 2, 1 VTSAY " Dois:" VTGET cVar2 WHEN RTRIM(cVar1) != "Um"
@ 3, 1 VTSAY " Tres:" VTGET cVar3 VALID !EMPTY(cVar3)
VTREAD
VTRestore(0,0,4,10,aTela)
```

VTRestore()

Tipo: TELNET VT100

Exibe uma região de tela gravada em uma localização especificada

Sintaxe

VTRESTORE(<nTopo>, <nEsquerda>,
<nBase>, <nDireita>, <aTela>) --> NIL

Parâmetros

<nTopo>, <nEsquerda>, <nBase>, e <nDireita> definem as coordenadas da informação de tela contida em <aTela>.
<aTela> , uma variável conteúdo o conteúdo da tela gravada.

Retorna

VTRESTORE () sempre retorna NIL.

Exemplo

```
#include 'apvt100.ch'
aTela := VTSave(0,0,4,10)
cVar1 := cVar2 := cVar3 := SPACE(10)
@ 1, 1 VTSAY " Um  :" VTGET cVar1 VALID !EMPTY(cVar1)
@ 2, 1 VTSAY " Dois:" VTGET cVar2 WHEN RTRIM(cVar1) != "Um"
@ 3, 1 VTSAY " Tres:" VTGET cVar3 VALID !EMPTY(cVar3)
VTREAD
VTRestore(0,0,4,10,aTela)
```

VTScroll()

Tipo: TELNET VT100

Rola uma região de tela para cima ou para baixo

Sintaxe

VTSCROLL(<nTopo>, <nEsquerda>,
<nBase>, <nDireita>, <nLinhas>) --> NIL

Parâmetros

<nTopo>, <nEsquerda>, <nBase>, e <nDireita> definem as coordenadas da regiao a ser rolada. Valores de linha e coluna podem variar entre 0, 0 e VTMAXROW(), VTMAXCOL().

<nLinhas> define a quantidade de linhas a serem roladas. Um valor maior do que zero rola para cima a quantidade especificada de linhas.

Um valor menor do que zero rola para baixo a quantidade especificada de linhas. Um valor de zero apaga a rea especificada.

Retorna

VTSCROLL() sempre retorna NIL.

Exemplo

```
VTSCROLL(0,0, VTMAXROW(),VTMAXCOL,1)
```

VTLastKey()

Tipo: TELNET VT100

Retorna o valor VTINKEY() da última tecla extraída do buffer de teclado

Sintaxe

VTLASTKEY() --> nCodInkey

Retorna

VTLASTKEY() retorna um n£mero de -39 a 386 que identifica o valor VTINKEY() da última tecla extraída do buffer de teclado.

Descrição

VTLASTKEY() , uma função de tratamento de teclado que informa o valor VTINKEY() da última tecla capturada do buffer de teclado pela função VTINKEY(), ou por um estado de espera como VTREAD, VTPAUSE, VTACHOICE(),VTABROWSE ou VTDBBROWSE. VTLASTKEY() retorna seu valor corrente
até que outra tecla seja capturada do buffer de teclado.

Exemplo

```
#include 'apvt100.ch'
aTela := VTSave(0,0,4,10)
cVar1 := cVar2 := cVar3 := SPACE(10)
@ 1, 1 VTSAY " Um  :" VTGET cVar1 VALID !EMPTY(cVar1)
@ 2, 1 VTSAY " Dois:" VTGET cVar2 WHEN RTRIM(cVar1) != "Um"
@ 3, 1 VTSAY " Tres:" VTGET cVar3 VALID !EMPTY(cVar3)
VTREAD
VTRestore(0,0,4,10,aTela)
If VTLastKey() == 27
  Return .f.
Endif
```

VTSetKey()

Tipo: TELNET VT100

Atribui um bloco de ação a uma tecla

Sintaxe

VTSETKEY(<nCodInkey>, [<bA‡ao>]) --> bAcaoCorrente

Parâmetros

<nCodInkey> , o valor INKEY() da tecla a ser associada ou questionada.
<bAcao> especifica o bloco de código a ser automaticamente executado sempre que a tecla especificada for pressionada durante um estado de espera.

Retorna

VTSETKEY() retorna o bloco de ação correntemente associado a tecla especificada, ou NIL caso a tecla especificada nao esteja associada a um bloco.

Exemplo

```
#include 'apvt100.ch'
bKeyAnt := VTSetKey(65,{|| teste()})
CCodigo := space(6)
@ 1, 1 VTGET cCodigo
VTREAD
VTSetKey(65,bKeyAnt)

RETURN

FUNCTION Teste()
  @ 2,1 VTSay 'TESTE'
RETURN NIL
```

VTKeyBoard()

Tipo: TELNET VT100

Coloca uma cadeia de caracteres (string) no buffer de teclado

Sintaxe

VTKEYBOARD(<cCodigoTecla>)

Parâmetros

<cCodigoTecla> , o conjunto de caracteres a ser colocado no buffer de teclado.

Exemplo

```
#include 'apvt100.ch'
bKeyAnt := VTSetKey(65,{|| teste()})
CCodigo := space(6)
@ 1, 1 VTGET cCodigo
VTREAD
VTSetKey(65,bKeyAnt)

RETURN

FUNCTION Teste()
  @ 2,1 VTSay 'TESTE'
   VTKeyBoard(chr(27))
RETURN NIL
```

VTRow()

Tipo: TELNET VT100

Retorna a posição de linha do cursor na tela

Sintaxe

VTROW() --> nLinha

Retorna

VTROW() retorna a posição de linha do cursor na forma de um valor numérico inteiro. A faixa do valor de retorno varia entre zero e VTMAXROW().

Exemplo

```
@ 0,0 VTSay "Teste"
@ VTRow()+1 ,0 VTSay "Teste2"
```

VTCol()

Tipo: TELNET VT100

Retorna a posição de coluna do cursor na tela

Sintaxe

VTCOL() --> nCol

Retorna

VTCOL() retorna um valor numérico inteiro. A faixa do valor de retorno é de zero até VTMAXCOL().

Exemplo

```
@ 1, 1 VTSAY "Cliente: " + TRIM(Cliente)
@ VTROW(), VTCOL() + 1 VTSAY  Status
```

VTInkey()

Tipo: TELNET VT100

Extrai um caractere do buffer de teclado

Sintaxe

VTINKEY([<nSegundos>]) --> nCodInkey

Parâmetros

<nSegundos> especifica a quantidade de segundos que VTINKEY() deve esperar por uma tecla. O valor pode ser especificado em incrementos do tamanho de até um décimo de segundo. Se for especificado zero, o programa para até, que uma tecla seja pressionada. Caso <nSegundos>

seja omitido, VTINKEY() não espera por uma tecla.

Retorna

VTINKEY() retorna um valor numérico inteiro de -39 at, 386, que identifica a tecla extraída do buffer de teclado. Caso o buffer de teclado esteja vazio, VTINKEY() retorna zero.

Exemplo

```
While .t.
    IF VtInkey(1) == 27  // correspondente a tela ESC
        exit
    EndIf
End
```

VTMaxCol()

Tipo: TELNET VT100

Determina a coluna máxima visível na tela

Sintaxe

VTMAXCOL() --> nColuna

Retorna

VTMAXCOL() retorna o número da coluna visível mais a direita para fins de exibição.

Exemplo

```
@ 1, int(VTMaxCOL()/2) VTSAY "*"
```

VTMaxRow()

Tipo: TELNET VT100

Determina a máxima linha visível na tela

Sintaxe

VTMAXROW() --> nLinha

Retorna

VTMAXROW() retorna o número da última linha visível para fins de exibição.

Descrição

VTMAXROW() , uma função de tratamento de tela que pode ser utilizada para determinar a m xima linha visível da tela. N£meros de linha e coluna
come‡am com zero em Clipper.

Exemplo

A seguinte função definida pelo usuário, TamTela(), utiliza VTMAXROW() e VTMAXCOL() para retornar um vetor que contém o tamanho da tela corrente:

```
FUNCTION TamTela
  RETURN { VTMAXROW(), VTMAXCOL() }
```

VTBeep()

Tipo: TELNET VT100

Emite um beep

Sintaxe

VTBEEP([<nQtde>]) --> NIL

Parâmetros

<nQtde> especifica a quantidade de beep que será emitido, Caso <nQtde> seja omitido, VTBEEP() emitira um beep.

Retorna

VTBEEP() retorna NIL

Exemplo

VTBEEP(3)

VTReverso()

Tipo: TELNET VT100

Ativa ou desativa o modo reverso da tela.

Sintaxe

VTREVERSO([<lRev>]) --> lReverso

Parâmetros

<lRev> Se verdadeiro ativa, falso desativa o modo de tela. Caso <lRev> seja omitido, VTReverso() retorna o modo atual.

Retorna

VTREVERSO() retorna o modo atual, verdadeiro que está em reverso, falso não está em reverso.

Exemplo

```
#include 'apvt100.ch'

lReverso:=  VTReverso(.t.)
@ 0,0 VTSay "Teste 1"
VTReverso(lReverso)
@ 1,0 VTSay "Teste 2"
```

VTClearBuffer()

Tipo: TELNET VT100

Limpa o buffer de teclado

Sintaxe

VTCLEARBUFFER() --> NIL

Retorna

VTCLEARBUFFER() retorna NIL

Exemplo

```
VTCLEARBUFFER()
```

VTAlert()

Tipo: TELNET VT100

Mostra uma messagem na tela

Sintaxe

VTALERT(<cMsg>,[<cCaption>],[<lCenter>],[<nSleep>]) →nTecla

Parâmetros

<cMsg> Mensagem a ser exibida.

<cCaption>Titulo da mensagem

<lCenter> Se verdadeiro centraliza a mensagem conforme o VTSetSize.

<nSleep> > especifica a quantidade de tempo em milésimo de segundos em a mensagem permanecera na tela, Caso omito, aguardara que seja digitado Enter ou ESC.

Retorna

VTALERT () retorna o código da tecla digitada.

Exemplo

VTAlert(“Produto não cadastrado”,”Aviso”,.t.,4000)

VTYesNo()

Tipo: TELNET VT100

Mostra uma messagem a espera de uma confirmação na tela.

Sintaxe

VTYESNO (<cMsg>,[<cCaption>],[<lCenter>]) →lConfrime

Parâmetros

<cMsg> Mensagem a ser exibida.
<cCaption>Titulo da mensagem
<lCenter> Se verdadeiro centraliza a mensagem conforme o VTSetSize.

Retorna

VTYESNO () retorna o verdadeiro caso tenha confirmado.

Exemplo

```
IF !  VTYesNo('Confirma a alteracao','Atencao ',.T.)
   Return .F.
ENDIF
```

VTAchoice()

Tipo: TELNET VT100

Executa um menu pop-up

Sintaxe

```
VTACHOICE(<nTopo>, <nEsquerda>, <nBase>, <nDireita>,
                <acItensMenu>, [<alItensSelecionaveis>],
               [<cFuncaoUsuario>],[<nItemInicial>], [lNaoBranco] , [<lMsg>],
               [<nLinhaJanela>],[<lScroll>]) --> nPosicao
```

Parâmetros

<nTopo>, <nEsquerda> e <nBase>, <nDireita> são as coordenadas do canto superior esquerdo e canto inferior direito da janela. Valores de linha podem variar entre zero e VTMAXROW(), e valores de coluna podem variar entre zero e VTMAXCOL().

<acItensMenu> é um vetor que contem as cadeias de caracteres que serão exibidas como sendo os itens de menu. Cada item de menu será mais tarde identificado através de sua posição numérica neste vetor.

<alItensSelecionaveis> é um vetor paralelo de valores lógicos, diretamente relacionados a <acItensMenu> que especifica os itens de menu que poderão ser selecionados. Os elementos podem ser valores lógicos ou cadeias de caracteres. Caso o elemento seja uma cadeia de caracteres, ele é avaliado como uma expressão macro que deverá retornar um tipo de dado lógico. Em ambos os casos, um valor de falso (.F.) significa que o item de menu correspondente não está disponível, e um valor de verdadeiro (.T.) significa que está disponível.

<cFuncaoUsuario> é o nome de uma função definida pelo usuário que é executada quando uma tecla não reconhecível for pressionada. O nome da função é especificado como uma expressão caractere sem parênteses ou argumentos. Note que o

comportamento de VTACHOICE() é afetado pela presença deste argumento. Consulte o texto abaixo para maiores informações.

<nItemInicial> é a posição ocupada no vetor de <acItensMenu> pelo item que aparecer em destaque quando o menu for exibido pela primeira vez. Caso você especifique um item de menu que não esteja disponível, ou caso você não use argumento algum, o item que aparecer em destaque será o primeiro item selecionável do vetor.
<lNaoBranco> Se for verdadeiro a opções do menu será montado conforme o tamanho da opção desconsiderando os espaços em branco à direita e esquerda. Caso seja negativa ou omitida a opção do menu será montado conforme a dimensão da tela do VTACHOICE definida em <nEsquerda> e <nDireita>.

<lMsg> Conteúdo tem que ser NIL, parâmetro reservado para implementação futura.

<nLinhaJanela> É o número da linha da janela na qual o item de menu inicial aparecerá.

Retorna

VTACHOICE() retorna a posição numérica ocupada pelo item de menu selecionado no vetor de <acItensMenu>. Se o processo de seleção for interrompido, VTACHOICE() retorna zero.

Função de usuário: Da mesma forma que as demais funções de interface com o usuário, VTACHOICE() aceita uma função de usuário. A função de usuário é especificada quando você deseja aninhar invocações da função VTACHOICE() para criar menus hierárquicos ou redefinir teclas.

Modos de VTACHOICE()
0-Inativo
1-Tentativa de passar início da lista
2-Tentativa de passar final da lista
3-Normal
4-tens não selecionados

Após a função de usuário ter executado as operações apropriadas ao modo VTACHOICE(), ela deve retornar um valor que solicite ao VTACHOICE() executar uma operação entre o seguinte conjunto de ações:

Valores de Retorno da Função de Controle de VTACHOICE()
0-Aborta seleção
1-Executa seleção
2-Continua VTACHOICE()
3-Vai para o próximo item cuja primeira letra for a tecla pressionada

Exemplo

```
acMenuItens := {“Um”,”Dois”, "-----", “Tres”}
alSelectableItens := {.T., .T., .F., .T.}
nPosition := VTACHOICE(0, 0, 7, 19, acMenuItens, alSelectableItens,”TESTECTRL” )

Function testectrl(modo,nElem,nElemW)
If modo == 1
        VtAlert('Top')
Elseif Modo == 2
        VtAlert('Bottom')
Else
        If VTLastkey() == 27
                VtAlert('sair')
                VTBeep(3)
                return 0
        elseIf VTLastkey() == 13
                VtAlert('ok')
```

```
                VtBeep(1)
                return 1
        Endif
EndIf
Return 2
```

VTABrowse()

Tipo: TELNET VT100

Monta um browse com referencia a um array.

Sintaxe

VTABROWSE(<nTopo>, <nEsquerda>, <nBase>, <nDireita>,
<aCab>, [<aItens>], [<aSize>],
[<cFuncaoUsuario>],[<nItemInicial>]) --> nPosicao

Parâmetros

<nTopo>, <nEsquerda> e <nBase>, <nDireita> são as coordenadas do canto superior esquerdo e canto inferior direito da janela. Valores de linha podem variar entre zero e VTMAXROW(), e valores de coluna podem variar entre zero e VTMAXCOL().

<aCab>, é um vetor que contem os títulos das colunas
<aItens>, é um vetor que contem os dados a serem mostrados
<aSize> , é um vetor que contem o tamanho de cada coluna

<cFuncaoUsuario> é o nome de uma função definida pelo usuário que é executada quando uma tecla não reconhecível for pressionada. O nome da função é especificado como uma expressão caractere sem parênteses ou argumentos. Note que o comportamento de VTABROWSE() é afetado pela presença deste argumento. Consulte o texto abaixo para maiores informações.

<nItemInicial> é a posição ocupada no vetor de < aItens > pelo item que aparecer em destaque quando o menu for exibido pela primeira vez. Caso você especifique um item de menu que não esteja disponível, ou caso você não use argumento algum, o item que aparecer em destaque será o primeiro item selecionável do vetor.

Retorna

VTABROWSE() retorna a posição numérica ocupada pelo item de menu selecionado no vetor de <aItens>. Se o processo de seleção for interrompido, VTABROWSE() retorna zero.

Função de usuário: Utilizada da mesma forma que VTACHOICE.

Modos de VTABROWSE()
0-Inativo
1-Tentativa de passar início da lista
2-Tentativa de passar final da lista
3-Normal
4-Itens não selecionados

Após a função de usuário ter executado as operações apropriadas ao modo VTABROWSE() ela deve retornar um valor que solicite ao VTABROWSE() executar uma operaçãao entre o seguinte conjunto de ações:

Valores de Retorno da Função de Controle de VTABROWSE()

0-Aborta seleção
1-Executa seleção
2-Continua VTABROWSE()
3-Vai para o próximo item cuja primeira letra for a tecla pressionada

Exemplo

```
#INCLUDE 'APVT100.CH'
VTClear
acab :={"Codigo","Cod          ","Descricao                    ","UM"}
aSize   := {10,4,20,10}
nPos := 12
aItens :={{"1010 ",10, "DESCRICAO1","UN "},;
        {"2010 ",20,"DESCRICAO2","CX "},;
        {"2020 ",30,"DESCRICAO3","CX "},;
        {"2010 ",40,"DESCRICAO4","CX "},;
        {"2020 ",50,"DESCRICAO5","CX "},;
        {"3010 ",60,"DESCRICAO6","CX "},;
        {"3020 ",70,"DESCRICAO7","CX "},;
        {"3030 ",80,"DESCRICAO7","CX "},;
        {"3040 ",90,"DESCRICAO7","CX "},;
        {"2010 ",40,"DESCRICAO4","CX "},;
        {"2020 ",50,"DESCRICAO5","CX "},;
        {"3010 ",60,"DESCRICAO6","CX "},;
        {"3020 ",70,"DESCRICAO7","CX "},;
        {"3030 ",80,"DESCRICAO7","CX "},;
        {"3050 ",100,"DESCRICAO7","CX "}}
npos := VTaBrowse(0,0,7,15,aCab,aItens,aSize,'testectrl',nPos)
.
.
.


Function testectrl(modo,nElem,nElemW)
If modo == 1
        VtAlert('Top')
Elseif Modo == 2
        VtAlert('Bottom')
Else
        If VTLastkey() == 27
                VtAlert('sair')
                VTBeep(3)
                return 0
        elseIf VTLastkey() == 13
                VtAlert('ok')
                VtBeep(1)
                return 1
        Endif
EndIf
Return 2
```

VTDBBrowse()

Tipo: TELNET VT100

Monta um browse com referencia a uma tabela

Sintaxe

VTDBBROWSE(<nTopo>, <nEsquerda>, <nBase>, <nDireita>,<cAlias>, <aCab>, [<aFields>], [<aSize>], [<cFuncaoUsuario>],[<cTop>],[<cBottom>]) --> nRecno

Parâmetros

<nTopo>, <nEsquerda> e <nBase>, <nDireita> são as coordenadas do canto superior esquerdo e canto inferior direito da janela. Valores de linha podem variar entre zero e VTMAXROW(), e valores de coluna podem variar entre zero e VTMAXCOL().

<cAlias>, é uma string com alias da tabela

<aCab>, é um vetor que contem os títulos das colunas
<aFields>, é um vetor que contem os campos do alias
<aSize> , é um vetor que contem o tamanho de cada coluna

<cFuncaoUsuario> é o nome de uma função definida pelo usuário que é executada quando uma tecla não reconhecível for pressionada. O nome da função é especificado como uma expressão caractere sem parênteses ou argumentos. Note que o comportamento de VTDBBROWSE () é afetado pela presença deste argumento. Consulte o texto abaixo para maiores informações.
<cTop> string com a condição de validação de top
<cBottom> string com a condição de validação de Bottom

Retorna

VTDBBROWSE () retorna o recno() Se o processo de seleção for interrompido, VTDBBROWSE () retorna zero.

Função de usuário: Utilizada da mesma forma que VTACHOICE e VTaBROWSE.

Modos de VTDBBROWSE ()
0-Inativo
1-Tentativa de passar início da lista
2-Tentativa de passar final da lista
3-Normal
4-Itens não selecionados

Após a função de usuário ter executado as operações apropriadas ao modo VTDBBROWSE () ela deve retornar um valor que solicite ao VTDBBROWSE () executar uma operação entre o seguinte conjunto de ações:

Valores de Retorno da Função de Controle de VTDBBROWSE ()
0-Aborta seleção
1-Executa seleção
2-Continua VTDBBROWSE ()
3-Vai para o próximo item cuja primeira letra for a tecla pressionada

Exemplo

```
VtClear
aFields := {"B1_COD","B1_DESC","B1_UM","B1_PICM"}
aSize   := {16,20,10,15}
aHeader := {'COD','DESCRICAO     ','UM',"% ICM"}

sb1->(dbseek(xfilial()+'00000000000001'))
nRecno := VTDBBrowse(0,0,7,15,"SB1",aHeader,aFields,aSize,'testectrl',;
                                          "xfilial('SB1')+'00000000000001'",;
                                          "xfilial('SB1')+'00000000000002'")
vtclear()
```

Exemplo aplicação Telnet VT100

```
#include "protheus.ch"
#include "apvt100.ch"

/*
Programa exemplo para Radio Freqüência
*/
Function AICDV035(xLocal,xLocaliz)
Local cEtiq := Space(TamSx3('CB0_CODET2')[1])
Local lWhen := .t.
Private cLocal
Private cLocaliz
```

```
Private lLocaliz := (GetMV("MV_LOCALIZ") == "S")
Private nLin:= 0

IF xLocal # '  '  .and. xLocal # NIL
        cLocal   := xLocal
        cLocaliz := xLocaliz
        lWhen := .f.
EndIf

While .t.
        nLin:= 0
        If lWhen
                cLocal   := Space(2)
                cLocaliz := Space(20)
        EndIf
        VTClear()
        @ 0,0 VTSay "Inventario"
        @ 1,0 VTSay "Almoxarifado:"
        @ 2,0 VTGet cLocal pict '@!' VALID aiv035Loc() when lwhen
        If lLocaliz
                @ 3,0 VTSay "Localizacao:"
                @ 4,0 VTGet cLocaliz VALID aiv035Locz() when lwhen
                nLin := 5
        Else
                nLin := 3
        EndIf
        VTRead
        If VTLastKey() == 27  // CASO O USUARIO PRECIONE ESC OU CLEAR
                Exit
        Endif
        If ! aiv035CriaCBA() .OR. ! aiv035CBB()
                If ! lWhen
                        exit
                Else
                        Loop
                EndIf
        EndIf
        @ nLin++,0 VTSay "Etiqueta        "
        @ nLin++,0 VTGet cEtiq VALID aiv035CBC(cEtiq)
        VTRead
        IF VTLastKey() == 27
                aiv035Fim()
        EndIf
        If ! lWhen
                exit
        EndIf
EndDo
Return

//FUNCAO DE VALIDACAO DA LOCALIZACAO
Static Function aiv035Loc()
If Empty(cLocal)
        VTBeep(3)       //EMITE UM BEEP NO R.F.
        VTAlert('Almoxarifado nao pode ser em branco!',"Aviso",.T.,2000)  //MOSTRA A
MENSAGEM
        VTKeyBoard(chr(20))    // LIMPA O GET ATIVO CTRL -T
```

```
        Return .F.
Endif
Return .T.
.
.
.
.
```

FUNCOES PARA MICROTERMINAL.

TerCls()

Tipo: Microterminal

Apaga todos os caracteres existentes na tela no microterminal preenchendo a tela com espaços em branco.

Sintaxe

TerCls()

Retorna

NIL

Exemplo

TerCls()

TerSay()

Tipo: Microterminal

Escreve no display do microterminal a string especificada pelo parâmetro <cMsg> na linha <nLin> e coluna <nCol>.

Sintaxe

TerSay(nLin,nCol,cMsg)

Parâmetros

nLin = Linha onde se deseja colocar a mensagem
nCol = Coluna onde se deseja colocar a mensagem
cMsg = Mensagem que se deseja colocar na tela

Retorna

NIL

Exemplo

TerSay(01,00,"Pressione <ENTER> para continuar.")

TerInkey()

Tipo: Microterminal

Especifica a quantidade de segundos que TerInkey() deve esperar por uma tecla. O valor pode ser especificado em incrementos de um segundo. Se for especificado zero, o programa para ate que uma tecla seja pressionada.

Sintaxe

TerInkey([nSegundos])

Parâmetros

nSegundos = Numero de segundos a aguardar

Retorna

Codigo da Tecla pressionada, se não foi pressionado nada o valor retornado e' ""

Exemplo

```
While !TerEsc()
    cTecla := TerInkey(0)
    TerSay(01,00,"Tecla Pressionada "+cTecla)
EndDo
```

TerCBuffer()

Tipo: Microterminal

Libera o buffer de teclado do microterminal, esse comando e' util em rotinas de interface com o usuario de forma a garantir que as teclas a serem processadas do buffer do teclado sao apropriadas a atividade corrente e nao pendentes de uma atividade anterior.

Sintaxe

TerCBuffer()

Retorna

Caracteres que estavam pendentes no buffer no teclado do microterminal.

Exemplo

```
TerCBuffer()
TerSay(01,00,"Pressione qualquer tecla para continuar.")
TerInkey(0)
```

TerGetRead()

Tipo: Microterminal

Executa uma entrada de dados no microterminal.

Sintaxe

TerGetRead(nLin,nCol,uVar,cPict,[bValid],[bWhen])

Parâmetros

nLin - Linha a se executar a entrada de dados.
nCol - Coluna a se executar a entrada de dados.
uVar - Variavel a se devolver o valor da entrada de dados.
cPict - Picture da entrada de dados, as pictures disponiveis sao:
"X" - Para entrada de caracteres numéricos(0-9) e letras (A-Z).
"9" - Para entrada de caracteres numéricos(0-9).
"A" - Para entrada de caracteres letra(A-Z).
"*" - Para entrada de caracteres numéricos(0-9) e letras (A-Z) mas exibindo "*" na tela.
"." - Exibe ponto decimal na tela.
"/" - Exibe a barra ("/") na tela.
bValid - Code Block contendo a validação da entrada de dados.
bWhen - Code Block contendo a condição para se executar a entrada de dados.

Retorna

NIL

Exemplo

```
lDigNome := .T.
cNome    := Space(10)
cIdade   := Space(02)
TerCls()
TerSay(00,00,"Nome:")
TerGetRead(00,06,@cNome,"XXXXXXXXXX",{||!Empty(cNome)},{||lDigNome})
TerGetRead(00,00,cIdade,"99",{||Val(cIdade)>0})
```

Observações

A variável da entrada de dados deve ser do tipo caracter.

TerEsc()

Tipo: Microterminal

Verifica se a ultima tecla pressionada no microterminal foi a DEL (consideramos como equivalente a <ESC> no teclado normal).

Sintaxe

TerEsc()

Retorna

.T. se a ultima tecla pressionada foi <DEL> ou .F. se não foi

Exemplo

```
TerCls()
While .T.
        cNome := Space(10)
        TerSay(00,00,"Nome:")
        TerGetRead(01,07,@cNome,"XXXXXXXXXX")
        If TerEsc()
                Exit
        EndIf
EndDo
TerCls()
TerSay(01,00,"Finalizando...")
```

TerBeep()

Tipo: Microterminal

Emite um sinal sonoro pelo 'buzzer' do Microterminal, caso o microterminal não possua este dispositivo instalado nada acontece.

Sintaxe

TerBeep([nVezes])

Parâmetros

nVezes - Numero de sinais sonoros a emitir.

Retorna

NIL

Exemplo

```
TerBeep(3)      // executa 3 sinais sonoros
TerBeep()       // executa apenas 1 sinal sonoro
```

TerNumTer()

Tipo: Microterminal

Retorna o numero do microterminal especificado no Monitor de microterminais.

Sintaxe

TerNumTer()

Retorna

Numero do microterminal

Exemplo

```
nTerminal := TerNumTer()
TerSay(00,00,"Terminal : "+StrZero(nTerminal))
```

TerSave()

Tipo: Microterminal

Grava uma regiao da tela do microterminal para posterior exibição.

Sintaxe

TerSave([nTopo],[nEsquerda],[nBase],[nDireita])

Parâmetros

nTopo - Linha inicial da região de tela a ser gravada.
nEsquerda - Coluna inicial da região de tela a ser gravada
nBase - Linha final da região de tela a ser gravada.
nDireita - Coluna final da região de tela a ser gravada.

Retorna

regiao de tela especificada na forma de cadeia de caracteres.

Exemplo

```
TerCls()
TerSay(00,00,"Tela a ser salva.")
cTela := TerSave(00,00,01,40)
TerCls()
TerRestore(00,00,01,40,cTela)
```

TerRestore()

Tipo: Microterminal

Restaura a região de tela do microterminal gravada pela função TerSave.

Sintaxe

TerRestore([nTopo],[nEsquerda],[nBase],[nDireita],cTela)

Parâmetros

nTopo - Linha inicial da região de tela a ser restaurada.
nEsquerda - Coluna inicial da região de tela a ser restaurada.
nBase - Linha final da região de tela a ser restaurada.
nDireita - Coluna final da região de tela a ser restaurada.
cTela - E' uma cadeia de caracteres que contem a região de tela gravada.

Retorna

NIL

Exemplo

```
TerCls()
TerSay(00,00,"Tela a ser salva.")
cTela := TerSave(00,00,01,40)
TerCls()
TerRestore(00,00,01,40,cTela)
```

TerPBegin()

Tipo: Microterminal

Executa função para inicio de impressão na porta paralela ou serial do microterminal.

Sintaxe

TerPBegin([nTerm],cSerPar)

Parâmetros

nTerm -

Numero do microterminal a iniciar a impressão, se nada for especificado será considerado o numero definido no Monitor de Microterminais.

cSerPar -

Informa qual a saída utilizada para impressão, 'S' para utilizar a saída SERIAL ou 'P' para se utilizar a saída PARALELA.

Retorna

NIL

Exemplo

```
TerPBegin(,"S")
TerPrint("Testando a impressão na saída SERIAL")
TerPEnd()
TerPBegin(,"P")
TerPrint("Testando a impressão na saída PARALELA")
TerPEnd()
```

TerPrint()

Tipo: Microterminal

Envia a string especificada para a saída de impressão definida pela função TerPBegin.

Sintaxe

TerPrint(cString, [lSalta])

Parâmetros

cString - Cadeia de caracteres a enviar para a saída de impressão.
lSalta - Informe .F. para não saltar a linha na impressora após o envio de cString ou .T. para que se salte uma linha apos o envio de cString, se não especificado .T. e' assumido.

Retorna

NIL

Exemplo

```
TerPBegin(,"P")
TerPrint("Imprimindo uma linha e saltando para proxima linha")
TerPrint("Imprimindo a linha e nao saltando.",.F.)          // não salta a linha
TerPrint("Continuando da linha anterior.")
TerPEnd()
```

TerPEnd()

Tipo: Microterminal

Finaliza impressão iniciada pela função TerPBegin.

Sintaxe

TerPEnd()

Retorna

NIL

Exemplo

```
TerPBegin(,"S")
TerPrint("Testando a impressão na saída SERIAL")
TerPEnd()
```

TerIsQuit()

Tipo: Microterminal

Função utilizada em pontos do programa para se verificar se o Monitor esta tentando finalizar a aplicação do microterminal, deve ser utilizada em lugares do programa em que o fechamento da aplicação não cause transtornos, como por exemplo, apos a conclusão de uma venda.

Sintaxe

TerIsQuit()

Retorna

.F.

Exemplo

```
While .T.
    TerCls()
    TerSay(00,00,"Inicando Venda...")
      .
      .
      .
      .
       .
```

```
        TerSay(00,00,"Venda Finalizada...")
        TerIsQuit()     // caso o monitor esteja finalizando a aplicação somente
                        //neste ponto a finalização será efetuada, garantindo
                        //assim que a venda seja concluída.
EndDo
```

Exemplo aplicação Microterminal

```
#INCLUDE "PROTHEUS.CH"
/*/
 Exemplo parte de programação para microterminal
/*/
Function AICDT040(cImpCB)
Local cVolume
Local cEmb
Local cDisp
Local cProduto
Private cPedido
Private cCodEmb
Private cImp := cImpCB
Private nTamCB0 := 8
cImp := PadR(AllTrim(cImp),6)
If !CB5->(DbSeek(xFilial()+cImp))
        ConOut("Codigo da impressora do microterminal "+StrZero(TerNumTer(),2)+;
        " nao existe ("+cImp+")")   //
        Return
EndIf

While .t.
        TerIsQuit()
        cVolume  := Space(08)
        cEmb     := Space(03)
        cCodEmb  := Space(08)
        cDisp    := Space(03)
        cProduto := Space(08)
        TerCls()
        TerCBuffer()
        TerSay(00,00,"Volume:")
        TerSay(01,00,"Digite <ENTER> para novo volume.")
        TerGetRead(00,07,@cVolume,"XXXXXXXX",{||TVldVol(@cVolume,@cEmb)})
        If TerEsc()
                Loop
        EndIf
        TerCls()
        TerSay(00,00,"Volume:")
        TerSay(00,19,"Emb:")
        TerSay(00,07,cVolume)
        TerGetRead(00,23,@cEmb,"XXX",{|| TVldEmb(@cEmb,cVolume)},{||
Empty(cEmb)})
        If TerEsc()
                Loop
        EndIf
        TerSay(00,23,cEmb)
        TerSay(01,00,"Operador:")
        TerGetRead(01,09,@cCodEmb,"XXXXXXXX",{|| TVldCEmb()})
        If TerEsc()
                Loop
```

```
        EndIf
        TerCls()
        TerSay(00,00,"Volume:"+cVolume+"  Emb:"+cEmb+"  Oper:"+cCodEmb) //######
        TerSay(01,00,"Produto:") //
        TerGetRead(01,08,@cProduto,Replicate("X",08),{||
TVldProd(@cProduto,@cDisp,@cVolume)})
        TerCls()
        TerCBuffer()
        TerIsQuit()
EndDo
Return .T.

Static Function TVldVol(cVolume,cEmb,cVol2)
Local aRet:={}
Local cSavTel
Local cResBuff
Sleep(100)   // pausa para o buffer ser completado !!!
cResBuff := TerCBuffer()
If cResBuff == Chr(13)
        cResBuff := ""
EndIf
cVolume +=cResBuff

If cVol2 == NIL .and. !Empty(cVolume)
        aRet := CBRetEti(cVolume,"05")
        If len(aRet) == 0
                cSavTel := TerSave(01,00,01,40)
                TerSay(01,0,Padr("Etiqueta invalida!!!",40)) //
                TerBeep(4)
                Sleep(2000)
                TerRestore(01,00,01,40,cSavTel)
                Return .F.
        EndIf
        cVolume := aRet[1]
ElseIf cVol2 # NIL
        aRet := CBRetEti(cVol2,"05")
                If len(aRet) == 0
                TerSay(01,0,Padr("Etiqueta invalida!!!",40)) //
                TerBeep(4)
                Sleep(2000)
                TerSay(01,00,Padr("Confirme o Volume",40)) //
                Return .F.
        EndIf
        cVol2 :=aRet[1]
EndIf

If cVol2#NIL .and. cVolume#cVol2
        TerSay(01,0,Padr("Codigo de volume diferente!!!",40)) //
        TerBeep(4)
        Sleep(2000)
        TerSay(01,00,Padr("",40))
        Return .F.
EndIF

If Empty(cVolume)
        cVolume := Space(10)
```

```
        Return .t.
EndIf
/*
.
.
.
*/
Return .T.

Static Function TVldEmb(cEmb,cVolume)
If Empty(cEmb)
        TerSay(01,00,Padr("Embalagem nao pode ser branco.",40)) //
        TerBeep(4)
        Sleep(2000)
        TerSay(01,00,Space(40))
        Return .F.
EndIf
Return .T.
.
.
.
```