

# XHARBOUR – GTWVW

## DOCUMENTAÇÃO

### ÍNDICE GERAL

Introdução.....	005
Sobre a GTWVW.....	006
Sobre Janelas.....	007
Sobre Minimizar, Maximizar e Redesenhar Telas.....	008
Sobre o Cursor.....	009
Sobre o Espaçamento de Linhas.....	010
Sobre Fontes.....	011
Funções CALLBACK.....	012
Exemplos.....	013
Funções.....	
WVW_MakeDlgTemplate.....	014
WVW_AddRows.....	015
WVW_AppendMenu.....	016
WVW_CBAddString.....	017
WVW_CBCreate.....	018
WVW_CBDestroy.....	020
WVW_CBEnable.....	021
WVW_CBFindString.....	022
WVW_CBGetCurText.....	023
WVW_CBGetIndex.....	024
WVW_CBIsDropped.....	025
WVW_CBIsFocused.....	026
WVW_CBSetCodeBlock.....	027
WVW_CBSetCurSel.....	028
WVW_CBSetFont.....	029
WVW_CBSetIndex.....	030
WVW_CenterWindow.....	031
WVW_ChooseColor.....	032
WVW_ChooseFont.....	033
WVW_ClientToScreen.....	034
WVW_CreateDialogDynamic.....	035
WVW_CreateDialogModal.....	036
WVW_CreateFont.....	037
WVW_CreateMenu.....	038
WVW_CreatePopupMenu.....	039
WVW_CXCreate.....	040
WVW_CXDestroy.....	041
WVW_CXEnable.....	042
WVW_CXGetCheck.....	043
WVW_CXSetCheck.....	044
WVW_CXSetCodeBlock.....	045
WVW_CXSetFocus.....	046
WVW_DeleteMenu.....	047
WVW_DestroyMenu.....	048
WVW_DlgSetIcon.....	049
WVW_DrawBoxGet.....	050
WVW_DrawBoxGroup.....	051
WVW_DrawBoxRaised.....	052
WVW_DrawBoxRaised.....	053
WVW_DrawBoxRecessed.....	054
WVW_DrawButton.....	055
WVW_DrawColorRect.....	056
WVW_DrawEllipse.....	057
WVW_DrawFocusRect.....	058
WVW_DrawGridHorz.....	059

WVW_DrawGridVert.....	060
WVW_DrawImage.....	061
WVW_DrawLabel.....	062
WVW_DrawLabelEx.....	064
WVW_DrawLabelObj.....	065
WVW_DrawLine.....	066
WVW_DrawLineEx.....	068
WVW_DrawMenuBar.....	069
WVW_DrawOutline.....	070
WVW_DrawOutlineEx.....	072
WVW_DrawPicture.....	073
WVW_DrawProgressBar.....	074
WVW_DrawRectangle.....	074
WVW_DrawRoundRect.....	075
WVW_DrawScrollButton.....	076
WVW_DrawScrollThumbHorz.....	077
WVW_DrawScrollThumbVert.....	078
WVW_DrawShadedRect.....	079
WVW_DrawStatusBar.....	080
WVW_DrawTextBox.....	081
WVW_DrawToolButtonState.....	083
WVW_EnableMaximize.....	084
WVW_EnableMenuItem.....	085
WVW_EnableShortCuts.....	086
WVW_FillRectangle.....	087
WVW_GetClipboard.....	088
WVW_GetCursorPos.....	089
WVW_GetFontInfo.....	090
WVW_GetLastMenuEvent.....	091
WVW_GetMenu.....	092
WVW_GetPaintRect.....	093
WVW_GetPalette.....	094
WVW_GetRGBColor.....	095
WVW_GetRowColFromXY.....	096
WVW_GetScreenHeight.....	097
WVW_GetScreenWidth.....	098
WVW_GetTitle.....	099
WVW_GetTooltipBkColor.....	100
WVW_GetTooltipTextColor.....	101
WVW_getTooltipWidth.....	102
WVW_GetWindowHandle.....	103
WVW_GetXYFromRowCol.....	104
WVW_InvalidateRect.....	105
WVW_IsLButtonPressed.....	106
WVW_KillTimer.....	107
WVW_LBAddString.....	108
WVW_LBSetCursel.....	109
WVW_lCloseWindow.....	110
WVW_LoadFont.....	111
WVW_LoadPen.....	112
WVW_LoadPicture.....	113
WVW_Maximize.....	114
WVW_MaxMaxCol.....	115
WVW_MaxMaxRow.....	116
WVW_MessageBox.....	117
WVW_Minimize.....	118
WVW_nColOfs.....	119
WVW_nNumWindows.....	120
WVW_NoClose.....	121
WVW_nOpenWindow.....	122
WVW_NoStartupSubWindow.....	123
WVW_nRowOfs.....	124
WVW_nSetCurWindow.....	125
WVW_NumBMCache.....	126

WVW_PasteFromClipboard.....	127
WVW_PBCreate.....	128
WVW_PBDestroy.....	129
WVW_PBEnable.....	130
WVW_PBSetCodeBlock.....	131
WVW_PBSetFocus.....	132
WVW_PBSetFont.....	133
WVW_PBSetStyle.....	134
WVW_PGCreate.....	135
WVW_PGDestroy.....	136
WVW_PGGetPos.....	137
WVW_PGSetPos.....	138
WVW_ProcessMessages.....	139
WVW_Restore.....	140
WVW_RestScreen.....	141
WVW_SaveScreen.....	142
WVW_SBAddPart.....	143
WVW_SBCreate.....	144
WVW_SBDestroy.....	145
WVW_SBGetParts.....	146
WVW_SBGetText.....	147
WVW_SBRefresh.....	148
WVW_SBSetText.....	149
WVW_SetAltF4Close.....	150
WVW_SetAsNormal.....	151
WVW_SetBrush.....	152
WVW_SetClipboard.....	153
WVW_SetCodePage.....	154
WVW_SetDefCentreWindow.....	155
WVW_SetDefHCentreWindow.....	156
WVW_SetDefLineSpacing.....	157
WVW_SetDefLSpaceColor.....	158
WVW_SetDefVCentreWindow.....	159
WVW_SetFont.....	160
WVW_SetIcon.....	161
WVW_SetLastMenuEvent.....	162
WVW_SetLineSpacing.....	163
WVW_SetLSpaceColor.....	164
WVW_SetMainCoord.....	165
WVW_SetMaxBMCache.....	166
WVW_SetMenu.....	167
WVW_SetMenuKeyEvent.....	168
WVW_SetMouseMove.....	169
WVW_SetMousePos.....	170
WVW_SetOnTop.....	171
WVW_SetPaintRefresh.....	172
WVW_SetPalette.....	173
WVW_SetPen.....	174
WVW_SetPointer.....	175
WVW_SetPopupMenu.....	176
WVW_SetTimer.....	177
WVW_SetTitle.....	178
WVW_SetTooltip.....	179
WVW_SetTooltipActive.....	180
WVW_SetTooltipBkColor.....	181
WVW_SetTooltipMargin.....	182
WVW_SetTooltipText.....	183
WVW_SettooltipTextColor.....	184
WVW_SetTooltipTitle.....	185
WVW_SetTooltipWidth.....	186
WVW_SetVertCaret.....	187
WVW_SetWindowCentre.....	188
WVW_SetWindowPos.....	189
WVW_SetWinStyle.....	190

WVW_ShowWindow.....	191
WVW_TBAddButton.....	192
WVW_TBButtonCount.....	193
WVW_TBCMD2Index.....	194
WVW_TBCreate.....	195
WVW_TBDelButton.....	196
WVW_TBDestroy.....	197
WVW_TBEnableButton.....	198
WVW_TBIndex2CMD.....	199
WVW_TrackPopupMenu.....	200
WVW_UnreachedBR.....	201
WVW_UpdateWindow.....	202
WVW_XBCreate.....	203
WVW_XBDestroy.....	205
WVW_XBEnable.....	206
WVW_XBInfo.....	207
WVW_XBShow.....	208
WVW_XBUpdate.....	209
WVW_xReposWindow.....	210

## INTRODUÇÃO

Alguns agradecimentos são indispensáveis :

### **Harbour/xHarbour**

Claro, sem essa fantástica ferramenta todo esse trabalho não existiria ( os agradecimentos, na verdade, são para todas as pessoas que desenvolvem/contribuem ).

### **GTWVW**

O autor da GTWVW ( Budyanto Dj. ) e seus "ancestrais".

### **Marcos Antonio Gambeta**

Ele escreveu um guia de programação para a GTWVT e disponibilizou como freeware. A leitura desse manual me ajudou a resolver diversas dúvidas quanto aos parâmetros das funções, retorno de valores, etc.

### **Forum de notícias xHarbour**

Leitura obrigatória.

### **Forum Clipper on line**

Muita gente boa disposta a dividir seus conhecimentos.

### **MS SDK HELP FILES**

Leitura indispensável para entender realmente como e porquê de alguns parâmetros e características da GTWVW.

### **Julio C. Cantillo Molina**

Seu trabalho na WVVTOOLS abriu meus olhos para as reais possibilidades da GTWVW.

A realização desse trabalho de documentação das funções da GTWVW foi feito com muita atenção, mas nem por isso, está imune a falhas, erros de interpretação e gramaticais.

É a versão 0.01 alpha para português, cabe aos usuários da GTWVW fornecer correções, exemplos, trechos de códigos.

Criei um grupo no yahoo ( <http://br.groups.yahoo.com/group/gtwvw> ) para discussão das características técnicas, resolução de problemas, trocas de experiências e tudo mais relacionado com a GTWVW.

## **SOBRE A GTWVW**

GTWVW é um driver de terminal para xharbour com algumas bibliotecas em tempo de execução, permitindo ao programador mesclar texto e elementos GUI, em uma aplicação multi-janela. GTWVW é exclusivamente desenhado para a plataforma win32.

Usando a GTWVW o programador pode usar todas as funções padrão da GT, normalmente chamadas indiretamente pelo xharbour, como :

- ?, ?? (QOut(), QQOut())
- @ ... SAY ... (DevPos(), DevOut())
- Scroll()
- SetPos()
- \_GET\_()
- ReadModal()
- Inkey()
- AChoice()
- Alert()
- etc.

Todos os comandos e funções tem o mesmo comportamento que teriam em outras GTs ( por exemplo, no modo console ).

Podemos citar algumas características especiais da GTWVW :

- Permite ao programador realizar operações sobre janelas ( abrir, fechar, minimizar, maximizar, etc ).
- Pode mesclar elementos de texto e GUI em uma mesma janela.
- Controle nativos do windows ( statusbar, toolbars, scrollbars, pushbuttons, checkboxes ).

Veremos essas e outras características com detalhes mais adiante.

## SOBRE JANELAS

Algumas convenções básicas :

- As janelas são numeradas de 0..n, sendo 0 a janela principal, e n a janela atual.
- A janela principal é automaticamente aberta durante a inicialização do programa.
- Todas as janelas são automaticamente fechadas quando o programa termina.
- A janela-pai da janela n que será aberta é a janela atual ( tipicamente a janela n-1 ).
- A grande maioria das funções tem como primeiro parâmetro o número da janela atual, mas a GTWVW não utiliza esse parâmetro. Ao invés disso, a GTWVW "acha" a janela mais atual e utiliza-a. Mesmo assim, temos que considerar o número da janela para a ordem correta dos parâmetros. Por exemplo, a função para excluir um combobox é definida da seguinte forma

```
WVW_CBDestroy( nWinNum, nCBId )
```

onde **nWinNum** é o número da janela e **nCBId** o identificador do combobox. Para a GTWVW o parâmetro **nWinNum** é ignorado, então tanto faz chamarmos a função assim

```
WVW_CBDestroy( nWinNum, nCBId )
```

ou assim

```
WVW_CBDestroy( , nCBId )
```

## COORDENADAS

Existem dois modelos de coordenadas da tela, que o usuário pode selecionar/mudar a qualquer momento :

### **Standard Mode**

- Nesse modo as coordenadas são relativas a janela atual.
- A janela atual é sempre setada, no início do programa ou em cada operação de abertura e fechamento de janela.
- Todas as saídas/entradas da tela são orientadas para a janela principal. Dessa forma as funções maxrow() e maxcol() retornarão os limites da janela atual.

### **Maincoord Mode**

- Nesse modo as coordenadas são relativas a janela principal ( como no Clipper ).
- Todas as saídas/entradas trabalham baseadas na janela atual. Internamente, existe um processo que verifica em qual janela deve ser feita a operação de saída/entrada, dependendo da linha/coluna. Após cada operação, a janela atual é sempre resetada para a janela principal. Dessa forma maxrow() e maxcol() sempre retornarão as coordenadas máximas da janela principal, não importando o número de janelas abertas.
- Esse modo foi projetado para ser a forma mais rápida e eficiente para portar aplicativos do Clipper para xHarbour.

## SOBRE MINIMIZAR, MAXIMIZAR E REDESENHAR TELAS

Algumas funções da GTWVW não são automaticamente redesenhadas quando as janelas são minimizadas ou sobrepostas por outros elementos. Nossa aplicação deve “lembrar” quais são os elementos que devem ser redesenhados e a GTWVW nos ajuda com isso.

Existe uma função, `WVW_PAINT()`, definida pela nossa aplicação, que é chamada pela GTWVW, para que possamos redesenhar os nossos elementos gráficos ou qualquer outra coisa que quisermos.

Algumas das funções da GTWVW que precisam do suporte para redesenho são :

```
WVW_DrawBoxGet()  
WVW_DrawBoxRaised()  
WVW_DrawBoxRecessed()  
WVW_DrawBoxGroup()  
WVW_DrawImage()  
WVW_DrawLabel()
```

Observe que o redesenho não é feito de forma imediata pela GTWVW. O intervalo para redesenho pode ser definido pela nossa aplicação através da função `WVW_SetPaintRefresh()`.

Se setarmos o intervalo para redesenho para zero, a GTWVW chamará a função `WVW_PAINT()` cada vez que foi requisitado o redesenho, pelo windows ( exceto se uma chamada prévia ainda não foi retornada ).

Se o intervalo para redesenho for setado para maior que zero ( valores válidos maiores que 50 ), então a função `WVW_PAINT()` será chamada após esse intervalo, em milissegundos, apenas se ainda persistir uma ação de redesenho pendente.

O intervalo default para o redesenho é de 100.



## SOBRE O CURSOR

Existem dois estilos para o cursor :

Horizontal ( como em aplicações MS-DOS )

Vertical ( mais comum em aplicações Windows )

O programador pode selecionar qual o estilo que deseja, através da função **WVW\_SetVertCaret()**.

O novo estilo do cursor será aplicado para todas as janelas ( atualmente o cursor é apenas mostrado na janela atual ).

O estilo default é o horizontal.

## SOBRE O ESPAÇAMENTO DE LINHAS

O programador pode escolher se haverá espaço entre linhas. Isso pode desejável, entre outras razões, porque os elementos GUI podem sobrescrever a linha acima ou abaixo. Cada janela pode ter seu próprio espaçamento, que pode ser configurado através da função **WVW\_SetDefLineSpacing()**.

O espaçamento default é de zero.

## SOBRE FONTES

Nas diversas funções da GTWVW sobre as fontes, existem diversos parâmetros para controlar o tipo, largura, altura e outros fatores da fonte. Vamos ver aqui os detalhes de cada parâmetro da funções sobre fontes.

Por exemplo, na função **WVW\_CreateFont()**, tem a seguinte lista de parâmetros :

**cFontFace, nHeight, nWidth, nWeight, lItalic, lUnderline, lStrikeout, nCharset, nQuality, nEscapement**

e o significado para cada um é :

### **cFontFace**

Uma string que especifica o nome da fonte. O tamanho dessa string não deve exceder 32 caracteres.

### **nHeight**

A largura da fonte.

### **nWidth**

O tamanho da fonte.

### **nWeight**

Especifica o "peso" da fonte, variando de 0 até 1000. Por exemplo, um "peso" de 400 geralmente determina uma fonte normal, já 700, negrito. Existem 15 ( quinze ) modelos de "peso" de fonte, que podem ser encontradas no arquivo WINGDI.CH (FW\_DONTCARE, FW\_THIN, etc.).

### **lItalic**

Identifica se a fonte será itálica ou não.

### **lUnderline**

Identifica uma fonte sublinhada ou não.

### **lStrikeOut**

Identifica uma fonte *strikeout* ( fonte com linha traçada no meio das letras ).

### **nCharSet**

Especifica o conjunto de caracteres a ser usado. Por exemplo, ANSI\_CHARSET, DEFAULT\_CHARSET, OEM\_CHARSET, etc. ( os modelo estão no arquivo WINGDI.CH ).

### **nQuality**

Identifica a qualidade de saída da fonte. Essa característica define como a GDI deve interpretar a forma lógica da fonte, com a sua forma de apresentação física. Existem três valores possíveis para a qualidade da fonte :

<b>DEFAULT_QUALITY</b>	A aparência da fonte não importa.
<b>DRAFT_QUALITY</b>	Aparência intermediária de qualidade da fonte.
<b>PROOF_QUALITY</b>	A qualidade da fonte é a melhor forma possível.

### **nEscapement**

Especifica o ângulo, em décimos de graus, entre o vetor de fuga e o eixo-x do dispositivo. O vetor de fuga é paralelo com a linha base do texto. O default é 0.

## FUNÇÕES CALLBACK

Existem algumas funções que devem ser definidas na nossa aplicação, que são na verdade, chamadas diretamente pela GTWVW. Algumas das principais funções são :

### **WVW\_PAINT( nWinNum )**

Essa função é chamada sempre que o Windows recebe uma mensagem WM\_PAINT ( para redesenho da tela ). Na verdade o intervalo para chamada de WVW\_PAINT() pode ser configurado, através da função **WVW\_SetPaintRefresh()**.

### **WVW\_TIMER( nWinNum, hWnd, message, wParam, lParam )**

Essa função é chamada a cada intervalo de tempo, que pode ser definido através da função **WVW\_SetTimer()**.

### **WVW\_SIZE( nWindow, hWnd, message, wParam, lParam )**

Chamada sempre que a janela é minimizada, maximizada ou restaurada. Em conjunto com essa função, deve ser definida a função **WVW\_Size\_Ready()**, que indica se o processamento de **WVW\_Size()** deve ser realizado ou não.

## **EXEMPLOS**

No site do grupo tem uma pequena aplicação de exemplo do uso da GTWVW, algumas extensões, exemplos de tela, arquivo manifest, etc.

## FUNÇÃO

```
WVW__MakeDlgTemplate( Dlg,aItens1,aItens2,aItens3,aItens4,aItens5,aItens6,aItens7,aItens8,  
aItens9, aItens10 )
```

## PARÂMETROS

aDlg

aItens1

aItens2

aItens3

aItens4

aItens5

aItens6

aItens7

aItens8

aItens9

aItens10

## DESCRIÇÃO

Cria um modelo de janela de diálogo ( template ).

## RETORNO

String com o novo template de diálogo criado e o seu tamanho.

## FUNÇÃO

**WVW\_AddRows ( nWinNum, nRows )**

## PARÂMETROS

**nWinNum**

Número da janela. O Default é a janela corrente.

**nRows**

Número de linhas para adicionar ou diminuir ( se nRows < 0 ).

## DESCRIÇÃO

Adiciona ou diminui **nRows** linhas da janela **nWinNum**.

As novas linhas adicionadas serão coloridas com a cor da coluna 0 da linha anterior.

Não haverá checagem para determinar se a janela se tornou maior que a área de desktop (exceto se no modo MainCoord, porque precisamos desses limites para a função MAXROW()).

## RETORNO

.T. se a operação foi bem-sucedida.

## FUNÇÃO

**WVW\_AppendMenu( hMenu, nFlags, nMenuItemId, cCaption )**

## PARÂMETROS

### **hMenu**

identifica o menu que será incluído o item.

### **nFlags**

Especifica os flags para controlar a aparência e o comportamento do item que está sendo incluído. Esse parâmetro pode ser uma combinação de diversos valores ( as definições dos flags válidos estão no arquivo **WINUSER.CH** )

### **hMenuItemId**

Especifica cada um dos identificadores do novo item do menu ou, se o parâmetro **nFlags** estiver setado para MF\_POPUP, o handle do submenu.

### **cCaption**

Especifica ou a string que será apresentada no menu/submenu ou o caminho para o bitmap que será apresentado no menu/submenu.

## DESCRIÇÃO

Essa função, adiciona um novo item de menu para o menu **hMenu**. Através dessa função podemos também especificar o conteúdo, aparência e o comportamento de cada item do menu.

## RETORNO

Se a função for bem-sucedida, retorna um valor diferente de zero, caso contrário retorna zero.



## **FUNÇÃO**

**WVW\_CBAddString( nHandleDialog, nIDComboBox, cItem )**

## **PARÂMETROS**

### **nHandleDialog**

Handle da janela de diálogo.

### **nIDComboBox**

Identificador do combobox.

### **cItem**

Item que deve ser incluído no combobox.

## **DESCRIÇÃO**

Adiciona um item em um combobox.

## **RETORNO**

nenhum.

## FUNÇÃO

**WVW\_CBCreate( nWinNum, nTop, nLeft, nWidth, aText, bBlock, nListLines, nReserved, nKbdType, aOffset )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft**

Coordenadas para o combobox.

### **nWidth**

Tamanho do combobox.

### **aText**

Matriz com os elementos do combobox. O default é { "empty" }.

### **bBlock**

Bloco de código que será executado para os seguintes eventos :

**CBN\_SELCHANGE:** Usuário mudou a seleção ( não é executado quando a mudança é feita pelo programa )  
**CBN\_SETFOCUS:** Quando o foco vai para o combobox  
**CBN\_KILLFOCUS:** Quando o foco sai do combobox

Esse bloco de código será avaliado com os seguintes parâmetros :

<b>nWinNum</b>	Número da janela atual
<b>nCBId</b>	Identificador do combobox
<b>nType</b>	Tipo do evento ( apenas os três informados acima são suportados )
<b>nIndex</b>	Índice do item selecionado ( começando em 0 )

### **nListLines**

Número de itens que serão apresentados no combobox ( o default é 3 ). O número será automaticamente ajustado se esse parâmetro for maior que a matriz **aText**.

### **nReserved**

Reservado para uso futuro ( esse parâmetro atualmente é ignorado ).

### **nKbdType**

Especifica o comportamento do combobox. Pode ser um dos seguintes valores :

**0:** Similar as convenções dos programas para o windows ( isto é, ENTER/ESC faz com que o combobox perca o foco )  
**1:** Similar ao Clipper ( ENTER mostra os itens do combobox, UP/DOWN/TAB/SHIFTTAB/ESC faz o combobox perder o foco )

O default é **0**.

### **aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do combobox com as linhas/colunas.

### **DESCRIÇÃO**

Cria um combobox para a janela atual.

### **RETORNO**

O handle do combobox, se a operação for bem-sucedida, 0 se falhar.

## **FUNÇÃO**

**WVW\_CBDestroy( nWinNum, nCBId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nCBId**

Identificador do combobox.

## **DESCRIÇÃO**

Exclui um combobox da janela atual.

## **RETORNO**

Nenhum.

## **FUNÇÃO**

**WVW\_CBEnable( nWinNum, nCBId, lEnable )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nCBId**

Identificador do combobox.

**lEnable**

Habilita ( .T. ) ou desabilita ( .F. ) o acesso ao combobox.

## **DESCRIÇÃO**

Habilita/Desabilita o acesso ao combobox **nCBId** da janela atual.

## **RETORNO**

Retorna o estado anterior do combobox ( .T. habilitado, .F. desabilitado ). Se a função falhar, também retorna .F..

## FUNÇÃO

**WVW\_CBFindString( nWinNum, nCBId, cString )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nCBId**

Identificador do combobox.

**cString**

String para procurar.

## DESCRIÇÃO

Procura a string **cString** no combobox **nCBId** da janela atual.

## RETORNO

Índice do item que contém a string, ou -1 se falhar ( ou não encontrar ).

## **FUNÇÃO**

**WVW\_CBGetCurText( nWinNum, nCBId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janel atual.

**nCBId**

Identificador do combobox.

## **DESCRIÇÃO**

Obtém a string da posição atual do combobox.

## **RETORNO**

Retorna a string, se a operação for bem-sucedida, ou "" se falhar.

## **FUNÇÃO**

**WVW\_CBGetIndex( nWinNum, nCBId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nCBId**

Identificador do combobox.

## **DESCRIÇÃO**

Obtém a seleção atual em um combobox ( começando em 0 ).

## **RETORNO**

Retorna a posição atual em um combobox ou **-1** se falhar.



## **FUNÇÃO**

**WVW\_CBIsDropped( nWinNum, nCBId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nCBId**

Identificador do combobox.

## **DESCRIÇÃO**

Obtém o estado atual de um combobox.

## **RETORNO**

.T. se os itens estão sendo mostrados, .F. caso contrário.

## FUNÇÃO

**WVW\_CBISFocused( nWinNum, nCBId )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nCBId**

Identificador do combobox.

## DESCRIÇÃO

Obtém a informação se o foco está atualmente no combobox **nCBId** da janela atual.

## RETORNO

.T. se o foco está no combobox, .F. caso contrário.

## FUNÇÃO

**WVW\_CBSetCodeBlock( nWinNum, nCBId, bBlock )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nCBId**

Identificador do combobox.

**bBlock**

Novo bloco de código que será atribuído ao combobox.

## DESCRIÇÃO

Atribui um novo bloco de código para o combobox **nCBId** da janela atual.

## RETORNO

.T. se a operação foi bem-sucedida, .F. se falhar.

## **FUNÇÃO**

`WVW_CBSetCurSel( nHandleDialog, nIDCombobox, nIndex )`

## **PARÂMETROS**

### **nHandleDialog**

Handle da janela de diálogo.

### **nIDCombobox**

Identificador do combobox.

### **nIndex**

Índice do item, iniciando em 0. Se for -1, remove a seleção corrente e limpa o controle edit associado ao combobox.

## **DESCRIÇÃO**

Seleciona um item na lista de um combobox.

## **RETORNO**

nenhum.

## **FUNÇÃO**

**WVW\_CBSetFont( nWinNum, cFontFace, nHeight, nWidth, nWeight, nQuality, lItalic, lUnderline, lStrikeout )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **cFontName**

Nome da Fonte.

### **nHeight**

Largura da fonte.

### **nWidth**

Altura da fonte.

### **nWeight**

“Peso” da fonte.

### **nQuality**

Qualidade da fonte.

### **lItalic**

Indica se a fonte está em itálico ( .T. ) ou não ( default ).

### **lUnderline**

Indica se a fonte está sobrescrita ( .T. ) ou não ( default ).

### **lStrikeout**

Indica se a fonte está strikeout ( .T. ) ou não ( default ).

## **DESCRIÇÃO**

Inicializa a fonte para todos os combobox da janela atual ( e para todos os que serão criados posteriormente ).

## **RETORNO**

.T. se a operação for bem-sucedida, .F. se falhar.

## **FUNÇÃO**

**WVW\_CBSetIndex( nWinNum, nCBId, nIndex )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nCBId**

Identificador do combobox.

**nIndex**

Índice do item no combobox ( começando em 0 ).

## **DESCRIÇÃO**

Configura a seleção atual em um combobox.

## **RETORNO**

.T. se a operação foi bem-sucedida, .F. se falhar.

## FUNÇÃO

**WVW\_CenterWindow( nWinNum, lCenter, lPaint )**

Verifique a função **WVW\_SetWindowCentre()**.

## **FUNÇÃO**

**WVW\_ChooseColor( nRGBInit, aRGB16, nFlags )**

## **PARÂMETROS**

### **nRGBInit**

Cor inicial.

### **aRGB16**

Matriz com 16 elementos, contendo os índices das cores. O default é uma matriz com o conjunto de cores de um botão.

### **nFlags**

Flags que indicam a posição inicial no diálogo de seleção de cores. O default é a combinação das opções CC\_ANYCOLOR | CC\_RGBINIT | CC\_FULLOPEN.

## **DESCRIÇÃO**

Abre um diálogo de seleção de cores.

## **RETORNO**

Um valor RGB indicando a seleção de cor do usuário, ou zero se o usuário cancelar o diálogo.



## FUNÇÃO

**WVW\_ChooseFont( cFontName,nHeight,nWidth,nWeight,nQuality,lItalic,lUnderline,lStrikeout )**

## PARÂMETROS

**cFontName**

Nome da fonte.

**nHeight**

Largura da fonte.

**nWidth**

Altura da fonte.

**nWeight**

“Peso” da fonte.

**nQuality**

Qualidade da fonte.

**lItalic**

Indica se a fonte está em itálico ( .T. ) ou não ( default ).

**lUnderline**

Indica se a fonte está sobrescrita ( .T. ) ou não ( default ).

**lStrikeout**

Indica se a fonte está strikeout ( .T. ) ou não ( default ).

## DESCRIÇÃO

Abre um diálogo de seleção de fonte.

## RETORNO

Nenhum.

## **FUNÇÃO**

**WVW\_ClientToScreen( nWinNum, nRow, nCol )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nRow**

Linha da janela atual.

**nCol**

Coluna da janela atual.

## **DESCRIÇÃO**

Converte linha e coluna para as coordenadas x,y relativas ao vídeo.

## **RETORNO**

Matriz com duas posições contendo as coordenadas x,y.

## **FUNÇÃO**

**WVW\_CreateDialogDynamic(cDlgTemplate|nResource, lHandle, cDlgProc|bDlgProc|nDlgProc, nFlag)**

## **PARÂMETROS**

**cDlgTemplate|nResource**

**lHandle**

**cDlgProc|bDlgProc|nDlgProc**

**nFlag**

## **DESCRIÇÃO**

Cria uma janela de diálogo dinamicamente.

## **RETORNO**

Handle da janela de diálogo ou 0.

## **FUNÇÃO**

**WVW\_CreateDialogModal(cDialog|nResource|cDlgTemplate, NIL, bDlgProc|cDlgProc,nFlag,nHandle)**

## **PARÂMETROS**

**cDialog|nResource|cDlgTemplate**

**bDlgProc|cDlgProc**

**nFlag**

**nHandle**

## **DESCRIÇÃO**

Cria uma janela de diálogo modal.

## **RETORNO**

Handle da janela de diálogo ou 0.

## **FUNÇÃO**

`WVW_CreateFont(cFontFace, nHeight, nWidth, nWeight, lItalic, lUnderline, lStrikeout, nCharset, nQuality, nEscapement )`

## **PARÂMETROS**

### **cFontFace**

Uma string que especifica o nome da fonte.

### **nHeight**

A largura da fonte.

### **nWidth**

O tamanho da fonte.

### **nWeight**

“Peso” da fonte.

### **lItalic**

Identifica que a fonte será itálica.

### **lUnderline**

Identifica uma fonte sublinhada.

### **lStrikeOut**

Identifica uma fonte *strikeout*.

### **nCharSet**

Especifica o conjunto de caracteres a ser usado.

### **nQuality**

Identifica a qualidade de saída da fonte.

### **nEscapement**

Especifica o ângulo.

## **DESCRIÇÃO**

Cria uma fonte lógica com as características informadas nos parâmetros. Essa fonte pode ser selecionada como a fonte corrente para todos os dispositivos usados pela aplicação.

## **RETORNO**

Handle da fonte se a operação de criação for bem-sucedida, 0 ( zero ) caso contrário.

## **FUNÇÃO**

**WVW\_CreateMenu()**

## **PARÂMETROS**

nenhum

## **DESCRIÇÃO**

Cria um menu. Inicialmente o menu está vazio, mas pode ser preenchido por itens através da função `WVW_AppendMenu()`.

## **RETORNO**

Retorna o handle do novo menu criado. Se falhar, retorna NULL.

## **FUNÇÃO**

**WVW\_CreatePopupMenu()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Cria um menu drop-down, submenu ou menu de atalho. O menu está inicialmente vazio, mas pode ser preenchido através da função WVW\_AppendMenu().

## **RETORNO**

Retorna o handle do novo menu popup criado. Se falhar, retorna NULL.

## FUNÇÃO

**WVW\_CXCreate( nWinNum, nTop, nLeft, nBottom, nRight, cText, cImage/nImage, bBlock, aOffset, nStretchBitmap, lMap3DColors )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para criação do checkbox.

### **cText**

Texto do checkbox.

### **cImage/nImage**

Se o parâmetro for numérico, indica um ID de um RESOURCE em um arquivo .RC.

Se o parâmetro for uma string, indica um arquivo de imagem ( deve ser informado o caminho completo ).

### **bBlock**

Bloco de código que será executado sempre que um evento BN\_CLICK for executado. Sempre serão passados os parâmetros de **nWinNum** e o identificador do checkbox para a função.

Esse parâmetro tem que ser informado, caso contrário o checkbox não é criado.

### **aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do checkbox com as linhas/colunas.

### **nStretchBitmap**

Um número entre 0 e 1 ( inclusive ) como um fator para "esticar" o bitmap. Por exemplo :

<b>1.0</b>	O bitmap vai cobrir por inteiro o botão
<b>0.5</b>	O bitmap vai cobrir 50% do botão
<b>0</b>	O bitmap não vai ser "esticado"

O default é 1.

### **lMap3DColors**

Se a imagem terá efeito de transparência ( .T. ) ou não ( .F. ). Atenção para o fato que existe a limitação de bitmaps com, no máximo, 256 cores.

## DESCRIÇÃO

Cria um checkbox.

## RETORNO

Se a operação for bem-sucedida, retorna o handle do checkbox. Se falhar, retorna 0.



## **FUNÇÃO**

**WVW\_CXDestroy( nWinNum, nCXId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nCXId**

Identificador do checkbox.

## **DESCRIÇÃO**

Exclui um checkbox da janela atual.

## **RETORNO**

Nenhum.

## FUNÇÃO

**WVW\_CXEnable( nWinNum, nCXId, lToggle )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nCXId**

Identificador do checkbox.

**lToggle**

Habilita/Desabilita um checkbox.

## DESCRIÇÃO

Permite habilitar/desabilitar um checkbox **nCXId** na janela atual.

## RETORNO

Retorna o estado anterior do checkbox, se o parâmetro **lToggle** for informado, caso contrário, retorna o estado atual do checkbox ( .T. indica habilitado, .F. desabilitado ).

## **FUNÇÃO**

**WVW\_CXGetCheck( nWinNum, nCXId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nCXId**

Identificador do checkbox

## **DESCRIÇÃO**

Obtém o estado atual de um checkbox.

## **RETORNO**

Pode retornar os seguintes valores :

<b>0</b>	Não selecionado
<b>1</b>	Selecionado
<b>2</b>	Indeterminado

## **FUNÇÃO**

**WVW\_CXSetCheck( nWinNum, nCXId, nCheckState )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nCXId**

Identificador do checkbox.

**nCheckState**

**0**      Não selecionado

**1**      Selecionado

**2**      Indeterminado

## **DESCRIÇÃO**

Muda o estado de um checkbox.

## **RETORNO**

.T.

## FUNÇÃO

**WVW\_CXSetCodeBlock( nWinNum, nCXId, bBlock )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nCXId**

Identificador do checkbox.

**bBlock**

Bloco de código.

## DESCRIÇÃO

Atribui um novo bloco de código **bBlock** ao checkbox **nCXId** na janela atual.

## RETORNO

.T. se a operação for bem-sucedida, .F. se falhar.

## **FUNÇÃO**

**WVW\_CXSetFocus( nWinNum, nCXId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nCXId**

Identificador do checkbox.

## **DESCRIÇÃO**

Configura o foco para o checkbox **nCXId**.

## **RETORNO**

.T. se a operação foi bem-sucedida, .F. se falhar.

## FUNÇÃO

**WVW\_DeleteMenu( hMenu, nPosition, nFlag )**

## PARÂMETROS

### **hMenu**

Identifica o menu que será alterado.

### **nPosition**

Especifica o item do menu que será deletado, como determinado pelo parâmetro **nFlag**.

### **nFlag**

Especifica como o parâmetro **nPosition** será interpretado. Esse parâmetro deve ser apenas um desses dois valores :

<b>MF_BYCOMMAND</b>	Indica que <b>nPosition</b> dá a identificação do item do menu ( default )
<b>MF_BYPOSITION</b>	Indica que <b>nPosition</b> dá a posição numérica ( começando por zero ) do item do menu.

## DESCRIÇÃO

Essa função deleta um item do menu especificado. Se o item do menu for a identificação de um submenu, todo o submenu será apagado.

## RETORNO

Se a função for bem-sucedida, retorna um valor diferente de zero, caso contrário retorna zero.

## FUNÇÃO

**WVW\_DestroyMenu( hMenu )**

## PARÂMETROS

**hMenu**

Identifica o menu que será destruído.

## DESCRIÇÃO

Essa função destroi o menu especificado por **hMenu**

## RETORNO

Se a função for bem-sucedida, retorna um valor diferente de zero, caso contrário retorna zero.



## **FUNÇÃO**

**WVW\_DlgSetIcon( nHandleDialog, ncIcon )**

## **PARÂMETROS**

**nHandleDialog**

Handle da janela de diálogo.

**ncIcon**

Se o parâmetro for passado como numérico, identifica o RESOURCE em um arquivo RC.

Se o parâmetro for passado como string, identifica o nome do arquivo que contém o ícone.

## **DESCRIÇÃO**

Define o ícone em uma janela de diálogo.

## **RETORNO**

Se a operação for bem-sucedida, o handle do ícone, caso contrário NIL.

## **FUNÇÃO**

**WVW\_DrawBoxGet( nWinNum, nRow, nCol, nWidth, aOffset )**

## **DESCRIÇÃO**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nRow, nCol**

Linha/Coluna para desenho do quadro da entrada de dados.

### **nWidth**

Tamanho do quadro.

### **aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do quadro de get com as linhas/colunas.

## **DESCRIÇÃO**

Desenha um quadro para a entrada de dados. Essa função desenha linhas brancas na parte externa do lado direito e da parte de baixo ( diferentemente da GTWVT ).

## **RETORNO**

.T.

## **FUNÇÃO**

**WVW\_DrawBoxGroup( nWinNum, nTop, nLeft, nBottom, nRight, aOffset )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom ,nRight**

Coordenadas para desenho do quadro.

**aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do quadro com as linhas/colunas.

## **DESCRIÇÃO**

Desenha um quadro.

## **RETORNO**

.T.

## FUNÇÃO

**WVW\_DrawBoxRaised( nNumWin, nTop, nLeft, nBottom, nRight, lTight/aOffset )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas que serão usadas para desenhar o quadro.

**lTight/aOffset**

Se o parâmetro **lTight** for usado, o quadro é desenhado dentro da região dos caracteres e as linhas da parte superior ( topo e esquerdo ) serão dois pixels abaixo para ajudar no espaçamento dos caracteres (esse tipo de objeto GUI geralmente é sobrescrito por caracteres).

Se o parâmetro **aOffset** for usado, uma matriz deve ser definida com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do quadro com as linhas/colunas.

## DESCRIÇÃO

Desenha um quadro em relevo nas coordenadas especificadas.

## RETORNO

.T.

## FUNÇÃO

**WVW\_DrawBoxRaised( nNumWin, nTop, nLeft, nBottom, nRight, lTight/aOffset )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas que serão usadas para desenhar o quadro.

**lTight/aOffset**

Se o parâmetro **lTight** for usado, o quadro é desenhado dentro da região dos caracteres e as linhas da parte superior ( topo e esquerdo ) serão dois pixels abaixo para ajudar no espaçamento dos caracteres (esse tipo de objeto GUI geralmente é sobrescrito por caracteres).

Se o parâmetro **aOffset** for usado, uma matriz deve ser definida com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do quadro com as linhas/colunas.

## DESCRIÇÃO

Desenha um quadro em relevo nas coordenadas especificadas.

## RETORNO

.T.

## FUNÇÃO

**WVW\_DrawBoxRecessed( nWinNum, nTop, nLeft, nBottom, nRight, lTight/aOffset )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas que serão usadas para desenhar o quadro.

**lTight/aOffset**

Se o parâmetro **lTight** for usado, o quadro é desenhado dentro da região dos caracteres e as linhas da parte superior ( topo e esquerdo ) serão dois pixels abaixo para ajudar no espaçamento dos caracteres (esse tipo de objeto GUI geralmente é sobrescrito por caracteres).

Se o parâmetro **aOffset** for usado, uma matriz deve ser definida com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do quadro com as linhas/colunas.

## DESCRIÇÃO

Desenha um quadro em baixo relevo nas coordenadas especificadas.

## RETORNO

.T.

## FUNÇÃO

**WVW\_DrawButton( nWinNum, nTop, nLeft, nBottom, nRight, cText, cImage/nImage, nFormat, nTextColor, nBkClor, nImageAt )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do botão.

### **cText**

Texto para o botão.

### **cImage/nImage**

Se o parâmetro for uma string, indica o caminho + nome do arquivo contendo uma imagem para ser lido.

Se o parâmetro for um número, indica a posição da imagem no cache de imagens do aplicativo.

### **nFormat**

Indica o formato do botão. Pode ser :

- 0**      Raised ( default )
- 1**      Recessed
- 2**      Outline

### **nTextColor**

Índice da cor do texto do botão

### **nBkColor**

Índice da cor de fundo do botão

### **nImageAt**

Parâmetro ignorado.

## DESCRIÇÃO

Desenha um botão.

## RETORNO

.T.

## **FUNÇÃO**

**WVW\_DrawColorRect( nWinNum, nTop, nLeft, nBottom, nRight, aOffPixels, nRGBColor )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do retângulo.

**aOffPixels**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do retângulo com as linhas/colunas.

**nRGBColor**

Cor para preenchimento do retângulo.

## **DESCRIÇÃO**

Desenha um retângulo colorido. Essa função é compatível com a **WVW\_FillRectangle()**, mantida apenas para compatibilidade com a GTWVT.

## **RETORNO**

nenhum.



## **FUNÇÃO**

**WVW\_DrawEllipse( nWinNum, nTop, nLeft, nBottom, nRight, aOffset )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas para desenho da elipse.

**aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento da elipse com as linhas/colunas.

## **DESCRIÇÃO**

Desenha uma elipse.

## **RETORNO**

Se bem-sucedido, retorna um valor não-zero, caso contrário retorna zero.

## **FUNÇÃO**

**WVW\_DrawFocusRect( nWinNum, nTop, nLeft, nBottom, nRight, aOffset )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do retângulo.

**aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do retângulo com as linhas/colunas.

## **DESCRIÇÃO**

Desenha um retângulo no estilo usado para indicar que ele está com o foco.

## **RETORNO**

Se bem-sucedido, retorna um valor não-zero, caso contrário retorna zero.

## **FUNÇÃO**

**WVW\_DrawGridHorz( nWinNum, nTop, nLeft, nBottom, nRight, nRows )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas que será desenhada a grade horizontal.

**nRows**

Número de linhas para a grade horizontal que será desenhada.

## **DESCRIÇÃO**

Desenha uma grade horizontal.

## **RETORNO**

.T.

## **FUNÇÃO**

**WVW\_DrawGridVert( nWinNum, nTop, nBottom, aCols, nCols, aOffset )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nBottom**

Linha inicial e final para desenho da grade vertical.

### **aCols**

Matriz que determina as colunas que serão desenhadas.

### **nCols**

Número de colunas que serão desenhadas.

### **aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento da grade vertical com as linhas/colunas.

## **DESCRIÇÃO**

Desenha uma grade vertical.

## **RETORNO**

.T.

## FUNÇÃO

**WVW\_DrawImage( nWinNum, nTop, nLeft, nBottom, nRight, cImage/nPictureSlot, lTight/aOffset, lTransparent )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho da imagem.

Se a coordenada **nBottom** é NIL, então a altura da imagem será proporcional a largura da imagem.

Se a coordenada **nRight** é NIL, então a largura da imagem será proporcional a altura da imagem.

Se ambas as coordenadas **nBottom** e **nRight** são NIL, então o tamanho original da imagem será usado.

### **cImage/nPictureSlot**

Se o parâmetro for passado como string, uma imagem será lida no caminho especificado por **cImage**.

Se o parâmetro for numérico, será um índice para a lista de imagens já carregada pelo aplicativo.

Aqui vale um comentário : toda imagem lida é colocada em um cache. Portanto é interessante não usar essa função para desenhar um grande número de imagens.

### **lTight/aOffset**

Indica como a imagem será ajustada em pixels para as coordenadas de linha/coluna.

### **lTransparent**

Se .T., indica que a image é transparente, .F. caso contrário.

Se for passado como .T., a cor para transparência usada será do pixel do canto superior esquerdo.

## DESCRIÇÃO

Desenha uma imagem.

## RETORNO

.T. se conseguiu desenhar a imagem corretamente, .F. caso contrário.

## FUNÇÃO

```
WVW_DrawLabel( nWinNum, nRow, nCol, cLabel, nAlign, nEscapement, nTextColor, nBkColor,  
               cFontFace, nHeight, nWidth, nWeight, nQuality, nCharSet, lItalic      ,  
               lUnderline, lStrikeOut )
```

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nRow, nCol**

Linha/Coluna para desenho do label.

### **cLabel**

String que será desenhada.

### **nAlign**

Alinhamento da string. O default é TA\_LEFT (as outras definições estão no arquivo WINGDI.CH).

### **nEscapement**

Especifica o ângulo.

### **nTextColor**

Índice da cor do label.

### **nBkColor**

Índice da cor de fundo do label.

### **cFontFace**

Nome da fonte.

### **nHeight**

Largura da fonte.

### **nWidth**

Altura da fonte.

### **nWeight**

“Peso” da fonte.

### **nQuality**

Identifica a qualidade de saída da fonte.

### **nCharSet**

Especifica o conjunto de caracteres a ser usado.

### **lItalic**

Identifica que a fonte será itálica.

### **lUnderline**

Identifica uma fonte sublinhada.

### **lStrikeOut**

Identifica uma fonte *strikeout*.

## **DESCRIÇÃO**

Desenha um label.

## **RETORNO**

.T. se conseguiu desenhar, .F. caso contrário.

## FUNÇÃO

**WVW\_DrawLabelEx( nWinNum, nRow, nCol, cLabel, nAlign, cTextColor, nBkColor, nSlotFont )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nRow, nCol**

Linha/Coluna para desenho do label.

### **cLabel**

String que será desenhada.

### **nAlign**

Alinhamento do texto no label. O default é TA\_LEFT ( as outras definições estão no arquivo WINGDI.CH ).

### **cTextColor**

Índice da cor para o texto.

### **nBkColor**

Índice da cor de fundo do label.

### **nSlotFont**

Índice do cache de fontes da aplicação. Se não existir uma fonte associada ao índice passado, a função não será executada.

## DESCRIÇÃO

Desenha um label.

## RETORNO

.T. se conseguiu desenhar a label, .F. caso contrário ( retornará falso também se o parâmetro **nSlotFont** for inválido ).



## FUNÇÃO

**WVW\_DrawLabelObj( nWinNum, nTop, nLeft, nBottom, nRight, cLabel, nAlignHorz, nAlignVert, nTextColor, nBkColor, hFont, aOffset )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenhar o label.

### **cLabel**

String que será desenhada.

### **nAlignHorz**

Alinhamento horizontal do label. O default é 0.

### **nAlignVert**

Alinhamento vertical do label. O default é 0.

### **nTextColor**

Índice da cor do label.

### **nBkColor**

Índice da cor do fundo do label.

### **hFont**

Fonte que será desenhada o label.

### **aOffset**

Matriz com quatro elementos de alinhamento do label com as coordenadas **nTop, nLeft, nBottom, nRight**.

## DESCRIÇÃO

Desenha um label.

## RETORNO

.T.

## **FUNÇÃO**

**WVW\_DrawLine( nWinNum, nTop, nLeft, nBottom, nRight, nOrient, nFormat, nAlign, nStyle, nThick, nColor, aOffset )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho da linha.

### **nOrient**

Orientação da linha :

0      para linha horizontal  
1      para linha vertical.

### **nFormat**

Formato da linha. Pode ser :

0      Raised  
1      Recessed  
2      Plain

### **nAlign**

Alinhamento da linha :

0      Center  
1      Top  
2      Bottom  
3      Left  
4      Right

### **nStyle**

Estilo do traço. As definições estão no arquivo WINGDI.CH ( PS\_SOLID, PS\_DOT, etc. ).

### **nThick**

Largura do traço. O default é **0**, que indica a largura de um único pixel.

### **nColor**

Cor da linha ( valor com 32 bits de uma cor RGB ).

### **aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento da linha com as linhas/colunas.

## **DESCRIÇÃO**

Desenha uma linha.

## **RETORNO**



## **FUNÇÃO**

**WVW\_DrawLineEx( nWinNum, nTop, nLeft, nBottom, nRight, nOrient, nFormat, nAlign, nSlotPen )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho da linha.

### **nOrient**

Orientação da linha :

- 0      para linha horizontal
- 1      para linha vertical.

### **nFormat**

Formato da linha. Pode ser :

- 0      Raised
- 1      Recessed
- 2      Plain

### **nAlign**

Alinhamento da linha :

- 0      Center
- 1      Top
- 2      Bottom
- 3      Left
- 4      Right

### **nSlotPen**

Índice do cache de traços.

## **DESCRIÇÃO**

Desenha uma linha.

## **RETORNO**

.T.

## **FUNÇÃO**

**WVW\_DrawMenuBar()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Redesenha o menu da janela atual.

## **RETORNO**

nenhum.

## FUNÇÃO

**WVW\_DrawOutline( nWinNum, nTop, nLeft, nBottom, nRight, nThick, nShape, nRGBColor )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do quadro.

### **nThick**

Formato do traço. As definições estão no arquivo WINGDI.CH ( PS\_SOLID, PS\_DOT, etc. ).

### **nShape**

Parâmetro ignorado.

### **nRGBColor**

Especifica a referência de cor para o traço ( valor com 32 bits de uma cor RGB ).

## DESCRIÇÃO

Desenha um quadro nas coordenadas **nTop,nLeft,nBottom,nRight**, com os parâmetros de estilo e cor especificados em **nThick** e **nRGBColor**.

## RETORNO

.T.

## **FUNÇÃO**

**WVW\_DrawOutlineEx( nWinNum, nTop, nLeft, nBottom, nRight, nSlotPen )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do quadro.

**nSlotPen**

Índice do cache de traços.

## **DESCRIÇÃO**

Desenha um quadro no formato outline.

## **RETORNO**

Nenhum.

## **FUNÇÃO**

**WVW\_DrawPicture( nWinNum, nTop, nLeft, nBottom, nRight, nSlot, lTight/aOffset )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas para desenho da imagem.

**nSlot**

Índice do cache de imagens.

**lTight/aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento da imagem com as linhas/colunas.

## **DESCRIÇÃO**

Desenha uma imagem, gravando-a no cache de imagens.

## **RETORNO**

Nenhum.



## FUNÇÃO

`WVW_DrawProgressBar( nWinNum, nTop, nLeft, nBottom, nRight, aOffPixels, nPercent, nBackColor, nBarColor, cImage, lVertical, nDirection )`

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop,nLeft,nBottom,nRight**

Coordenadas para desenho da barra de progresso.

### **aOffPixels**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento da barra de progresso com as linhas/colunas.

### **nPercent**

Valor que identifica o preenchimento total da barra de progresso.

### **nBackColor**

Cor de fundo.

### **nBarColor**

Cor da barra de progresso.

### **cImage**

Imagem que será apresentada quando a barra estiver em progresso.

### **lVertical**

.T. indica que a barra será vertical, .F. que será horizontal.

### **nDirection**

Pode ter dois valores :

- 0 Indica uma progressão da esquerda para direita
- 1 Indica uma progressão da direita para esquerda

## DESCRIÇÃO

Desenha uma barra de progresso.

## RETORNO

nenhum.

## **FUNÇÃO**

**WVW\_DrawRectangle( nWinNum, nTop, nLeft, nBottom, nRight, aOffset )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do retângulo

### **aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do retângulo com as linhas/colunas.

## **DESCRIÇÃO**

Desenha um retângulo.

## **RETORNO**

Se bem-sucedido, retorna um valor não-zero, caso contrário retorna zero.

## **FUNÇÃO**

**WVW\_DrawRoundRect(nWinNum, nTop, nLeft, nBottom, nRight, aOffset, nRoundHeight, nRoundWidth)**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do retângulo.

### **aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do retângulo com as linhas/colunas.

### **nRoundHeight**

Largura da elipse desenhada para arredondar os cantos do retângulo.

### **nRoundWidth**

Altura da elipse desenhada para arredondar os cantos do retângulo.

## **DESCRIÇÃO**

Desenha um retângulo com os cantos arredondados. O retângulo é desenhado com o estilo, cor do traço atual e pintando com o pincel atual.

## **RETORNO**

Se bem-sucedido, retorna um valor não-zero, caso contrário retorna zero.

## FUNÇÃO

`WVW_DrawScrollBar( nWinNum, nTop, nLeft, nBottom, nRight, aOffPixels, nTLBR, lDepressed )`

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do scrollbar.

### **aOffPixels**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do scrollbar com as linhas/colunas.

### **nTLBR**

Indica o sentido do botão de direção do scrollbar. Os seguintes valores são permitidos :

1      botão com seta para cima      :      ▲

2      botão com seta para esquerda :      ◀

3      botão com seta para baixo      :      ▼

4      botão com seta para direita    :      ▶

### **lDepressed**

Se .F., botão tem o desenho do tipo **raised**, caso contrário o modelo será **recessed**.

## DESCRIÇÃO

Desenha um scrollbar.

Com as funções `WVW_XB***()`, essa função não se tornou mais necessária.

## RETORNO

nenhum.

## **FUNÇÃO**

**WVW\_DrawScrollThumbHorz( nWinNum, nTop, nLeft, nBottom, nRight, aOffPixels, nThumbPos )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do botão de movimentação do scrollbar.

### **aOffPixels**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do botão de movimentação do scrollbar com as linhas/colunas.

### **nThumbPos**

Coluna para posicionamento inicial.

## **DESCRIÇÃO**

Desenha uma barra de rolagem para um scrollbar horizontal.

Com as funções **WVW\_XB\*\*\*()**, essa função não se tornou mais necessária.

## **RETORNO**

nenhum.

## **FUNÇÃO**

**WVW\_DrawScrollThumbVert( nWinNum, nTop, nLeft, nBottom, nRight, aOffPixels, nThumbPos )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do botão de movimentação do scrollbar.

### **aOffPixels**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do botão de movimentação do scrollbar com as linhas/colunas.

### **nThumbPos**

Linha para posicionamento inicial.

## **DESCRIÇÃO**

Desenha uma barra de rolagem para um scrollbar vertical.

Com as funções **WVW\_XB\*\*\*()**, essa função não se tornou mais necessária.

## **RETORNO**

nenhum.

## FUNÇÃO

**WVW\_DrawShadedRect( nWinNum,nTop,nLeft,nBottom,nRight,aOffPixels,nHorVert,aRGBb,aRGBe )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenhar o retângulo.

### **aOffPixels**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do retângulo com as linhas/colunas.

### **nHorVert**

Especifica o modo de desenho e como interpretar a matriz de preenchimento do retângulo - **aRGBb** e **aRGBe**. O default é **GRADIENT\_FILL\_RECT\_H** ( as outras definições possíveis estão em WINGDI.CH ).

### **aRGBb**

Matriz com quatro posições de cores ( Red, Green, Blue, Alpha ) da coordenada inicial do retângulo.

### **aRGBe**

Matriz com quatro posições de cores ( Red, Green, Blue, Alpha ) da coordenada final do retângulo.

## DESCRIÇÃO

Desenha um retângulo nas coordenadas indicadas nos parâmetros, com preenchimento de cores de forma gradiente, variando da cor de **aRGBb** à **aRGBe**.

## RETORNO

.T. se conseguiu desenhar, .F. caso contrário.

## **FUNÇÃO**

**WVW\_DrawStatusBar( nWinNum, nPanels, aPos )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nPanels**

Número de divisões da barra de status.

**aPos**

Matriz contendo as posições de linha/coluna para as divisões da barra de status.

## **DESCRIÇÃO**

Desenha uma barra de status.

Mantida apenas para compatibilidade com a GTWVT. Recomenda-se usar as funções **WVW\_SB\*\*\*()**.

## **RETORNO**

Nenhum.



## FUNÇÃO

**WVW\_DrawTextBox( nWinNum, nTop, nLeft, nBottom, nRight, aOffPixels, cText, nAlignHorz, nAlignVert, nTextColor, nBackColor, nBackMode, hFont )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do texto.

### **aOffPixels**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do texto com as linhas/colunas.

### **cText**

String com o texto que será desenhado.

### **nAlignHorz**

Especifica o alinhamento do texto dentro das coordenadas informadas. Pode ser :

- 0**      Alinhamento a esquerda
- 1**      Alinhamento a direita
- 2**      Alinhamento centralizado

O default é alinhamento a esquerda.

### **nAlignVert**

Parâmetro ignorado

### **nTextColor**

Cor do texto.

### **nBackColor**

Cor de fundo.

### **nBackMode**

O modo como a cor de fundo será usada. O default é OPAQUE ( outras definições no arquivo WINGDI.CH )

### **hFont**

Fonte usada para o texto.

## DESCRIÇÃO

Desenha um texto formatado no retângulo especificado pelas coordenadas dos parâmetros.

## RETORNO

nenhum.

## FUNÇÃO

`WVW_DrawToolButtonState( nWinNum, nTop, nLeft, nBottom, nRight, aOffPixels, nState )`

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para desenho do toolbutton.

### **aOffPixels**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do toolbutton com as linhas/colunas.

### **nState**

Estado do botão. Pode ser um desses valores :

- 0      Estilo do botão **flat**.
- 1      Estilo do botão **raised**.
- 2      Estilo do botão **recessed**.

## DESCRIÇÃO

Desenha um toolbutton.

## RETORNO

.T.

## **FUNÇÃO**

**WVW\_EnableMaximize( nWinNum, lEnable )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela corrente.

**lEnable**

Se .T., habilita o botão de maximizar. Se .F. desabilita.

## **DESCRIÇÃO**

Lê/Setar o botão de maximizar. Para poder habilitar o botão, a aplicação tem que ter a função callback WVW\_SIZE() definida.

## **RETORNO**

Retorna o estado anterior do botão de maximizar.

## FUNÇÃO

**WVW\_EnableMenuItem( hMenu, nPosition, nFlag )**

## PARÂMETROS

**hMenu**

Identifica o menu.

**nPosition**

Especifica o item do menu que será habilitado ou desabilitado, conforme o parâmetro **nFlag**.

**nFlag**

Na prática, para o xHarbour, deveremos definir os seguintes valores :

<b>MF_DISABLE</b>	Indica que o item do menu está desabilitado, não podendo ser selecionado.
<b>MF_ENABLE</b>	Indica que o item do menu está habilitado, portanto, pode ser selecionado.

## DESCRIÇÃO

Habilita ou desabilita itens do menu.

## RETORNO

Retorna a configuração anterior do item do menu. Se o item do menu não existir, retorna 0.

## **FUNÇÃO**

**WVW\_EnableShortCuts( nWinNum, lEnable )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**lEnable**

.T. habilita teclas de atalho.

.F. desabilita teclas de atalho.

## **DESCRIÇÃO**

Determina se as teclas de atalho estão habilitadas/desabilitadas para acesso aos itens do menu e do menu de sistema.

## **RETORNO**

Configuração anterior.

## FUNÇÃO

**WVW\_FillRectangle**(nWinNum,nTop,nLeft,nBottom,nRight, nRGBColor/hBrush, lTight, lUseBrush)

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas para preenchimento do retângulo.

**nRGBColor/hBrush**

Cor para preenchimento do retângulo ( valor com 32 bits de uma cor RGB ).

**lTight**

O preenchimento será ajustado para toda a área das coordenadas informadas.

**lUseBrush**

Se .T., o parâmetro **nRGBColor/hBrush** é interpretado como handle para um modelo de pincel, caso contrário é usado como cor RGB.

## DESCRIÇÃO

Preenche um retângulo usando ou uma cor ou um modelo de pincel.

## RETORNO

.T.

## **FUNÇÃO**

**WVW\_GetClipboard()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Obtém o texto da área de transferência.

## **RETORNO**

Texto da área de transferência.



## **FUNÇÃO**

**WVW\_GetCursorPos()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Obtém as coordenadas do cursor do mouse.

## **RETORNO**

Uma matriz com dois elementos, contendo a coordenada X e a coordenada Y, respectivamente.

## **FUNÇÃO**

**WVW\_GetFontInfo()**

## **PARÂMETROS**

nenhum

## **DESCRIÇÃO**

Obtém dados da fonte corrente.

## **RETORNO**

Matriz com sete elementos, na seguinte ordem :

- 1 - cFontFace        Nome da fonte ( por ex. Arial ).
- 2 - cFontHeight    Altura da fonte.
- 3 - nFontWidth     Largura da fonte.
- 4 - nFontWiegght   "Peso" da fonte.
- 5 - cFontQuality    Qualidade da fonte.
- 6 - PTEXTSIZE->x   Largura da fonte em pixels.
- 7 - PTEXESIZE->y   Tamanho da fonte em pixels.

## **FUNÇÃO**

**WVW\_GetLastMenuEvent()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Lê o último item do menu selecionado. Trabalha sempre na janela atual.

## **RETORNO**

Último item selecionado, ou 0 caso não tenha sido selecionado nenhuma opção anteriormente.

## **FUNÇÃO**

**WVW\_GetMenu()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Obtém o handle do menu da janela atual.

## **RETORNO**

Handle do menu.

## **FUNÇÃO**

**WVW\_GetPaintRect( nWinNum )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

## **DESCRIÇÃO**

Obtém as coordenadas da tela com pendência para redesenhar.

## **RETORNO**

Matriz com quatro elementos, contendo a linha/coluna inicial e a linha/coluna final, respectivamente.

## **FUNÇÃO**

**WVW\_GetPalette()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Obtém a paleta de cores.

## **RETORNO**

Matriz com 16 elementos, cada elemento representando uma cor.

## **FUNÇÃO**

**WVW\_GetRGBColor( nColor )**

## **PARÂMETROS**

**nColor**

Índice da cor ( como definido no Clipper ).

## **DESCRIÇÃO**

Lê o valor RGB do índice da cor passado no parâmetro **nColor**.

## **RETORNO**

Retorna o valor RGB da cor.

## **FUNÇÃO**

**WVW\_GetRowColFromXY( nWinNum, nX, nY )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nX**

Coordenada X em pixels.

**nY**

Coordenada Y em pixels.

## **DESCRIÇÃO**

Converte as coordenadas X,Y ( em pixels ) em linha e coluna.

## **RETORNO**

Matriz com dois elementos, contendo a linha e a coluna, respectivamente.



## **FUNÇÃO**

**WVW\_GetScreenHeight()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Lê a largura da tela, em pixels.

## **RETORNO**

A largura da tela, em pixels.

## **FUNÇÃO**

**WVW\_GetScreenWidth()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Lê o tamanho da tela, em pixels.

## **RETORNO**

Tamanho da tela, em pixels.

## **FUNÇÃO**

**WVW\_GetTitle()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Lê o título da janela atual.

## **RETORNO**

O título da janela atual.

## **FUNÇÃO**

**WVW\_GetTooltipBkColor()**

## **PARÂMETROS**

Nenhum.

## **DESCRIÇÃO**

Obtém a cor de fundo da tooltip.

## **RETORNO**

Índice de cor RGB com a cor de fundo da tooltip.

## **FUNÇÃO**

**WVW\_GetTooltipTextColor()**

## **PARÂMETROS**

Nenhum.

## **DESCRIÇÃO**

Obtém a cor do texto da tooltip.

## **RETORNO**

Índice de cor RGB com a cor do texto da tooltip.

## **FUNÇÃO**

**WVW\_GetTooltipWidth()**

## **PARÂMETROS**

Nenhum.

## **DESCRIÇÃO**

Obtém o tamanho do tooltip.

## **RETORNO**

Número com o tamanho do tooltip.

## **FUNÇÃO**

**WVW\_GetWindowHandle()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Obtém o handle da janela atual.

## **RETORNO**

Handle da janela.

## **FUNÇÃO**

**WVW\_GetXYFromRowCol( nWinNum, nRow, nCol )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nRow**

Número da linha.

**nCol**

Número da coluna.

## **DESCRIÇÃO**

Converte linha e coluna em coordenadas X,Y ( em pixels ).

## **RETORNO**

Matriz com dois elementos, contendo a coordenada X e a coordenada Y, respectivamente.



## **FUNÇÃO**

**WVW\_InvalidateRect()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Invalida a janela atual, forçando o seu redesenho.

## **RETORNO**

nenhum.

## **FUNÇÃO**

**WVW\_IsLButtonPressed()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Verifica se o botão esquerdo do mouse está pressionado.

## **RETORNO**

.T. se o botão estiver pressionado.  
.F. se o botão não estiver pressionado.

## FUNÇÃO

`WVW_KillTimer( nWinNum )`

## PARÂMETROS

`nWinNum`

Número da janela. O default é a janela atual.

## DESCRIÇÃO

Cancela o evento **timer** para a janela atual.

## RETORNO

.T. se conseguiu cancelar o evento **timer**, .F. caso contrário.

## **FUNÇÃO**

**WVW\_LBAddString( nHandleDialog, nIDListBox, cItem )**

## **PARÂMETROS**

**nHandleDialog**

Handle da janela de diálogo.

**nIDListBox**

Identificador do Listbox

**cItem**

Item para ser incluído no listbox.

## **DESCRIÇÃO**

Adiciona um item em um listbox.

## **RETORNO**

nenhum.

## **FUNÇÃO**

**WVW\_LBSetCurSel( nHandleDialog, nIDListBox, nItem )**

## **PARÂMETROS**

**nHandleDialog**

Handle da janela de diálogo.

**nIDListBox**

Identificador do listbox.

**nItem**

Índice do item, iniciando por 0.

## **DESCRIÇÃO**

Seleciona um item em um listbox.

## **RETORNO**

nenhum.

## **FUNÇÃO**

**WVW\_lCloseWindow()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Fecha a janela superior ( que está no topo, a última que foi criada ).

## **RETORNO**

.T. se conseguiu fechar.

## FUNÇÃO

```
WVW_LoadFont( nSlotFont, cFontName, nHeight, nWidth, nWeight, lItalic, lUnderline,  
              lStrikeout, nCharSet, nQuality, nEscapement )
```

## PARÂMETROS

### **nSlotFont**

Índice do cache de fontes.

### **cFontName**

Nome da Fonte.

### **nHeight**

Largura da fonte.

### **nWidth**

Altura da fonte.

### **nWeight**

“Peso” da fonte.

### **lItalic**

Indica se a fonte está em itálico ( .T. ) ou não ( default ).

### **lUnderline**

Indica se a fonte está sobrescrita ( .T. ) ou não ( default ).

### **lStrikeout**

Indica se a fonte está strikeout ( .T. ) ou não ( default ).

### **nCharSet**

Especifica o conjunto de caracteres a ser usado

### **nQuality**

Qualidade da fonte.

### **nEscapement**

Especifica o ângulo.

## DESCRIÇÃO

Carrega as informações da fonte especificada através dos seus parâmetros, gravando no cache de fontes.

## RETORNO

Nenhum.

## **FUNÇÃO**

**WVW\_LoadPen( nSlot, nStyle, nWidth, nRGBColor )**

## **PARÂMETROS**

**nSlot**

Índice do cache de traços definidos pelo usuário.

**nStyle**

Estilo do traço.

**nWidth**

Altura do traço. Se especificar 0, o traço terá apenas um pixel de extensão.

**nRGBColor**

A cor de referência para o traço.

## **DESCRIÇÃO**

Carrega as definições do traço ( estilo, tamanho e cor ) para o cache de traços.

## **RETORNO**

.T. se conseguiu ler e gravar as definições do traço, .F. caso contrário.



## **FUNÇÃO**

**WVW\_LoadPicture( nSlot, cFilePicture )**

## **PARÂMETROS**

**nSlot**

Posição no cache onde a imagem será carregada. Se já existir alguma imagem na posição informada, será substituída.

**cFilePicture**

Imagem a ser carregada no cache.

## **DESCRIÇÃO**

Carrega uma imagem no cache de imagens. Leia a observação sobre o cache de imagens, na introdução desse documento.

## **RETORNO**

.T. se conseguiu ler a imagem, .F. caso contrário.

## FUNÇÃO

**WVW\_Maximize( nWinNum )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

## DESCRIÇÃO

Maximiza a janela atual ( parâmetro **nWinNum** é ignorado ). Se a aplicação definir a função callback **WVW\_Size()** a janela será maximizada, caso contrário ela apenas será restaurada.

## RETORNO

nenhum.

## FUNÇÃO

**WVW\_MaxMaxCol ( nWinNum )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela corrente.

## DESCRIÇÃO

Retorna o máximo MAXCOL() possível na configuração de tela corrente, para a fonte usada na janela **nWinNum**

## RETORNO

Número máximo de colunas.

## FUNÇÃO

**WVW\_MaxMaxRow( nWinNum )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela corrente.

## DESCRIÇÃO

Retorna o máximo MAXROW() possível na configuração de tela corrente, para a fonte usada na janela **nWinNum**.

## RETORNO

Número máximo de linhas.

## **FUNÇÃO**

**WVW\_MessageBox( nWinNum, cMessage, cTitle, nOption )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **cMessage**

Mensagem que será exibida.

### **cTitle**

Título da janela de mensagem.

### **nOption**

Indica quais os botões serão usados na janela. O default é MB\_OK ( botão OK sem ícones ). As outras definições estão no arquivo WINUSER.CH.

## **DESCRIÇÃO**

Exibe uma janela de mensagem. A janela de mensagem pode ter um ícone e um ou mais botões com opções para o usuário escolher.

## **RETORNO**

O número da opção escolhida ( IDOK para o botão OK, IDCANCEL para o botão cancelar, etc. ). As definições para o retorno estão no arquivo WINUSER.CH.

## **FUNÇÃO**

**WVW\_Minimize()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Minimiza a janela atual.

## **RETORNO**

nenhum.

## FUNÇÃO

`WVW_nColOfs( nWinNum )`

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela corrente.

## DESCRIÇÃO

Determinar o número de colunas de offset da janela **nWinNum**, relativo a janela principal.

## RETORNO

Numero de colunas de offset.

## **FUNÇÃO**

**WVW\_nNumWindows()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Retorna o numero total de janelas que estão abertas ( incluindo a janela principal ).

## **RETORNO**

Numero de janelas abertas.



## FUNÇÃO

**WVW\_NoClose( nWinNum )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela corrente.

## DESCRIÇÃO

Desabilita o botão de 'X' da janela **nWinNum**.

## RETORNO

nenhum.

## **FUNÇÃO**

**WVW\_OpenWindow( cWinName, row1, col1, row2, col2, nStyle, nParentWin, nExStyle )**

## **PARÂMETROS**

### **cWinName**

É o título da janela. Se for omitido, será usado o nome do arquivo executável.

### **row1, col1, row2, col2**

Linha inicial, coluna inicial, linha final e coluna final da janela que será aberta. Essas coordenadas são relativas a janela principal e não a janela atual.

Essas coordenadas são usadas ainda para :

- 1) colocar a janela em seu posicionamento inicial.
- 2) determinar o tamanho da janela ( novo maxrow() e maxcol() ).
- 3) salvar em RowOfs e ColOfs para o modo de MainCoord.

### **nStyle**

É o estilo da janela ( por ex. WS\_OVERLAPPEDWINDOW, etc ). As definições estão em **WINUSER.CH**. O default é WS\_CAPTION|WS\_SYSMENU|WS\_CLIPCHILDREN. Se for usar na janela controles como PUSHBUTTON, você deve incluir o estilo WS\_CLIPCHILDREN.

### **nParentWin**

É a janela mãe da nova janela que está sendo aberta. O default é a janela atual ( no modo Standard ) e a última janela ( no modo MainCoord ). Se quiser que a nova janela não tenha mãe, informe -1.

### **nExStyle**

Estilo estendido para janela ( por exemplo WS\_EX\_TOOLTIPWINDOW ). Default é NIL.

## **DESCRIÇÃO**

Abre uma janela nas coordenadas especificadas. Dependendo do parâmetro da função WVW\_NoStartUpSubWindow() a janela será apresentada ou permanecerá invisível ( até que seja chamada a função WVW\_ShowWindow() ).

## **RETORNO**

O numero da janela se for bem sucedido ou 0 se falhar.

## **FUNÇÃO**

**WVW\_NoStartupSubWindow( lOn )**

## **PARÂMETROS**

**lOn**

.T. quando a janela for aberta, será apresentada ( default ).

.F. quando a janela for aberta, permanecerá invisível.

Se não for informado parâmetro, retorna configuração atual.

## **DESCRIÇÃO**

Informa ao sistema que a janela, quando criada, será aberta ( **lOn** = .T. ) ou permanecerá invisível ( **lOn** = .F. ) até que se chame a função WVW\_ShowWindow().

## **RETORNO**

Retorna a nova configuração para abertura de janelas ( se **lOn** for informado ) ou a configuração atual.

## FUNÇÃO

**WVW\_nRowOfs ( nWinNum )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela corrente.

## DESCRIÇÃO

Determinar o número de linhas de offset da janela **nWinNum**, relativo a janela principal.

## RETORNO

Numero de linhas de offset.

## FUNÇÃO

**WVW\_nSetCurWindow( nWinNum )**

## PARÂMETROS

**nWinNum**

A nova janela que receberá o foco ( se tornará a janela ativa ).

## DESCRIÇÃO

Tornar a **nWinNum** janela corrente ( em foco, ativa ). Tem lógica apenas para o modo StandadMode.

## RETORNO

A antiga janela corrente.

## **FUNÇÃO**

**WVW\_NumBMCache()**

## **PARÂMETROS**

Nenhum.

## **DESCRIÇÃO**

Obtém o número máximo do cache de bitmaps.

## **RETORNO**

Configuração atual do número máximo do cache de bitmaps.

## **FUNÇÃO**

**WVW\_PasteFromClipboard()**

## **PARÂMETROS**

nenhum

## **DESCRIÇÃO**

Cola o texto que está na área de transferência.

## **RETORNO**

nenhum.

## FUNÇÃO

```
WVW_PBCreate( nWinNum, nTop, nLeft, nBottom, nRight, cText, cImage/nImage, bBlock, aOffset,  
              nStretchBitmap, lMap3DColors )
```

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para criação do botão.

### **cText**

Texto para o botão. O default é "".

### **cImage/nImage**

Se informado como numérico indica o ID do RESOURCE em um arquivo .RC.  
Se informado como string, deve conter o caminho completo do arquivo de imagem.

### **bBlock**

Bloco de código que será executado sempre quando for gerado um evento BN\_CLICK ( ou seja, pressionar e liberar o botão ). Sempre será passado os parâmetros : **nWinNum** e o identificador do botão.

### **aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento do botão com as linhas/colunas.

### **nStretchBitmap**

Um número entre 0 e 1 ( inclusive ) como um fator para "esticar" o bitmap. Por exemplo :

<b>1.0</b>	O bitmap vai cobrir por inteiro o botão
<b>0.5</b>	O bitmap vai cobrir 50% do botão
<b>0</b>	O bitmap não vai ser "esticado"

O default é 1.

### **lMap3DColors**

Se a imagem terá efeito de transparência ( .T. ) ou não ( .F. ). Atenção para o fato que existe a limitação de bitmaps com, no máximo, 256 cores.

## DESCRIÇÃO

Cria um pushbutton na janela atual.

## RETORNO

Retorna o handle do novo pushbutton, se a operação for bem-sucedida, ou **0** se falhar.



## **FUNÇÃO**

**WVW\_PBDestroy( nWinNum, nPBId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nPBId**

Identificador do pushbutton.

## **DESCRIÇÃO**

Exclui um pushbutton da janela atual.

## **RETORNO**

Nenhum.

## **FUNÇÃO**

**WVW\_PBEnable( nWinNum, nPBId, lToggle )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nPBId**

Identificador do pushbutton.

**lToggle**

.T. Habilita pushbutton ( default )

.F. Desabilita pushbutton

## **DESCRIÇÃO**

Habilita/Desabilita pushbutton na janela atual.

## **RETORNO**

Retorna o estado anterior do pushbutton, se o parâmetro **lToggle** for informado, caso contrário, retorna o estado atual do pushbutton ( .T. indica habilitado, .F. desabilitado ).

## FUNÇÃO

**WVW\_PBSetCodeBlock( nWinNum, nPBId, bBlock )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nPBId**

Identificador do pushbutton.

**bBlock**

Bloco de código.

## DESCRIÇÃO

Atribui um novo bloco de código **bBlock** para o pushbutton **nPBId** na janela atual.

## RETORNO

.T. se a operação foi bem-sucedida, .F. se falhar.

## **FUNÇÃO**

**WVW\_PBSetFocus( nWinNum, nPBId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nPBId**

Identificador do pushbutton.

## **DESCRIÇÃO**

Configura o foco para o pushbutton da janela atual.

## **RETORNO**

.T. se conseguiu configurar o foco corretamente, .F. se falhar.

## **FUNÇÃO**

**WVW\_PBSetFont( nWinNum, cFontFace, nHeight, nWidth, nWeight, nQuality, lItalic, lUnderline, lStrikeout )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **cFontName**

Nome da Fonte.

### **nHeight**

Largura da fonte.

### **nWidth**

Altura da fonte.

### **nWeight**

“Peso” da fonte.

### **nQuality**

Qualidade da fonte.

### **lItalic**

Indica se a fonte está em itálico ( .T. ) ou não ( default ).

### **lUnderline**

Indica se a fonte está sobrescrita ( .T. ) ou não ( default ).

### **lStrikeout**

Indica se a fonte está strikeout ( .T. ) ou não ( default ).

## **DESCRIÇÃO**

Inicializa a fonte para os pushbuttons ( existentes e os que ainda serão criados ).

## **RETORNO**

.T. se a operação for bem-sucedida, .F. se falhou.

## FUNÇÃO

**WVW\_PBSetStyle( nWinNum, nPBId, nStyle )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nPBId**

Identificador do pushbutton.

**nStyle**

Define o estilo do pushbutton. As possíveis definições estão no arquivo WINUSER.CH ( BS\_PUSHBUTTON, BS\_DEFPUSHBUTTON, etc. ). O uso mais comum é através dos seguintes valores :

<b>BS_DEFPUSHBUTTON</b>	<b>1:</b> Indica que o pushbutton é o botão default ( aparece selecionado através de uma borda pontilhada ).
<b>BS_PUSHBUTTON</b>	<b>0:</b> Indica um pushbutton padrão.

## DESCRIÇÃO

Atribui um novo estilo ao pushbutton da janela atual.

## RETORNO

.T.

## **FUNÇÃO**

**WVW\_PGCreate( nWinNum, nTop, nLeft, nBottom, nRight, aOffset, nBackColor, nBarColor, lSmooth, lVertical )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nTop, nLeft, nBottom, nRight**

Coordenadas para a criação da progress bar.

### **aOffset**

Matriz com quatro elementos com coordenadas dos cantos superior, esquerdo, inferior e direito de offset para alinhamento da progress bar com as linhas/colunas.

### **nBackColor**

Cor de fundo.

### **nBarColor**

Cor da progress bar.

### **lSmooth**

.T. Desenha a progress bar como uma barra contínua

.F. Desenha a progress bar com pequenos retângulos distintos

### **lVertical**

.T. Desenha a progress bar vertical

.F. Desenha a progress bar horizontal

## **DESCRIÇÃO**

Desenha uma progress bar para a janela atual. A faixa da progress bar é inicialmente de 0 a 100, começando com 0.

## **RETORNO**

O handle da progress bar, se a operação for bem-sucedida, ou 0 se falhar.

## **FUNÇÃO**

**WVW\_PGDestroy( nWinNum, nPGId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nPGId**

Identificador da progress bar.

## **DESCRIÇÃO**

Exclui uma progress bar da janela atual.

## **RETORNO**

Nenhum.



## **FUNÇÃO**

**WVW\_PGGetPos( nWinNum, nPGId )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nPGId**

Identificador da progress bar.

## **DESCRIÇÃO**

Obtém a posição atual em uma progress bar.

## **RETORNO**

Posição na progress bar ou **0** se falhar.

## FUNÇÃO

**WVW\_PGSetPos( nWinNum, nPGId, nPos )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nPGId**

Identificador da progress bar.

**nPos**

Número da posição na progress bar ( dentro da faixa do mínimo-máximo ).

## DESCRIÇÃO

Atualiza a progress bar para a posição indicada pelo parâmetro **nPos**.

## RETORNO

.T. se a operação for bem-sucedida, .F. se falhar.

## **FUNÇÃO**

**WVW\_ProcessMessages()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Processa as mensagens de todas as janelas.

## **RETORNO**

nenhum ( na verdade retorna o número 1, mas sem nenhum utilidade ).

## FUNÇÃO

**WVW\_Restore( nWinNum )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

## DESCRIÇÃO

Restaura a janela atual ( o parâmetro **nWinNum** é ignorado ). A restauração da janela do modo maximizado pode precisar ser manipulada pela função callback **WVW\_Size()**, porque a função **WVW\_Restore()** assume que não acontecerá mudança em maxrow()/maxcol().

## RETORNO

nenhum.

## FUNÇÃO

**WVW\_RestScreen( nWinNum, nTop, nLeft, nBottom, nRight, aScr, lDoNotDestroyBMP )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas da tela onde serão apresentados os dados salvos através da função **WVW\_SaveScreen()**.

**aScr**

Matriz com três elementos contendo a altura, largura e handle do bitmap, respectivamente.

**lDoNotDestroyBMP**

.T. se após a restauração da tela, apagar o bitmap que contém a imagem da tela salva, .F. caso contrário.

## DESCRIÇÃO

Restaura a tela salva previamente através da função **WVW\_SaveScreen()**.

## RETORNO

.T. se conseguiu restaurar a tela com sucesso, .F. caso contrário.

## **FUNÇÃO**

**WVW\_SaveScreen( nWinNum, nTop, nLeft, nBottom, nRight )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas da região da tela que sera salva ( sempre da janela atual, pois o parâmetro **nWinNum** é ignorado ).

## **DESCRIÇÃO**

Salva uma região da tela, gravando em um bitmap.

## **RETORNO**

Matriz com três elementos, contendo o tamanho, largura e o handle do bitmap.

## FUNÇÃO

**WVW\_SBAddPart( nWinNum, cMaxText, nWidth, nStyle, lResetParts, cIcon, cTooltip )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **cMaxText**

Se definido esse parâmetro, o tamanho da parte da status bar será o tamanho do texto.

### **nWidth**

Tamanho da parte da status bar, em pixels. Se o parâmetro **cMaxText** for informado, **nWidth** será ignorado.

### **nStyle**

Estilo da status bar. Os valores permitidos são :

<b>0</b>	Recessed ( default )
<b>0x100</b>	Sem bordas
<b>0x200</b>	Raised

### **lResetParts**

Se .T., indica que todas as partes anteriores serão removidas, .F. indica que uma nova parte criada será adicionada as anteriores.

### **cIcon/cTooltip**

Parâmetros ignorados.

## DESCRIÇÃO

Adiciona uma divisão em uma status bar, com tamanho e estilo especificados nos parâmetros **cMaxText/nWidth** e **nStyle**.

## RETORNO

Retorna o número de partes atuais na status bar, se a operação for bem-sucedida, ou **0** caso contrário.

## **FUNÇÃO**

**WVW\_SBCreate( nWinNum )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

## **DESCRIÇÃO**

Cria um status bar para a janela corrente, com uma parte.

## **RETORNO**

Retorna um handle para a status bar, se a operação for bem-sucedida, ou **0** se falhar.



## **FUNÇÃO**

**WVW\_SBDestroy( nWinNum )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

## **DESCRIÇÃO**

Destrói um status bar, forçando a janela a ser redesenhada.

## **RETORNO**

Nenhum.

## **FUNÇÃO**

**WVW\_SBGetParts()**

## **PARÂMETROS**

Nenhum.

## **DESCRIÇÃO**

Lê o número de partes de uma status bar.

## **RETORNO**

Número de partes de uma status bar.

## FUNÇÃO

**WVW\_SBGetText( nWinNum, nPart )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nPart**

Número da parte da status bar.

## DESCRIÇÃO

Lê o texto da status bar contido na parte **nPart**.

## RETORNO

String da parte **nPart** da status bar.

### **FUNÇÃO**

**WVW\_SBRefresh()**

### **PARÂMETROS**

Nenhum.

### **DESCRIÇÃO**

Reinicializa todas as partes da status bar, redesenhado-as.

### **RETORNO**

Número de partes da status bar, se a operação for bem-sucedida, ou **0** caso contrário.

## FUNÇÃO

**WVW\_SBSetText( nWinNum, nPart, cText )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nPart**

Número da parte que o texto será configurado.

**cText**

Texto que será configurado para a parte **nPart** da status bar.

## DESCRIÇÃO

Configura um texto **cText** para a parte **nPart** de uma status bar.

## RETORNO

Nenhum.

## **FUNÇÃO**

**WVW\_SetAltF4Close( lOn )**

## **PARÂMETROS**

**lOn**

- .T. aplicação pode ser encerrada através de ALT + F4
- .F. aplicação não pode ser encerrada através de ALT + F4

## **DESCRIÇÃO**

Identifica se a aplicação pode ser encerrada através da combinação de tecla ALT + F4.

## **RETORNO**

Configuração anterior.

## **FUNÇÃO**

**WVW\_SetAsNormal()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Retorna a janela ao estado normal ( oposto da função **WVW\_SetOnTop()** ).

## **RETORNO**

.T. se conseguiu alterar a configuração, .F. caso contrário.

## **FUNÇÃO**

**WVW\_SetBrush( nStyle, nColor, nHatch )**

## **PARÂMETROS**

### **nStyle**

Identifica o estilo do pincel. As definições podem ser encontradas no arquivo WINGDI.CH ( BS\_SOLID, BS\_NULL, etc. ).

### **nColor**

Índice da cor para desenho do pincel.

### **nHatch**

Define o padrão de preenchimento do pincel. As definições estão no arquivo WINGDI.CH ( HS\_VERTICAL, HS\_HORIZONTAL, etc. ).

## **DESCRIÇÃO**

Define o estilo, cor e padrão de preenchimento do pincel.

## **RETORNO**

.T. se conseguiu definir corretamente, .F. caso contrário.



## **FUNÇÃO**

**WVW\_SetClipboard( cText )**

## **PARÂMETROS**

**cText**

Texto que será inserido na área de transferência.

## **DESCRIÇÃO**

Configura texto para a área de transferência.

## **RETORNO**

.T. se conseguir configurar, caso contrário retorna .F..

## **FUNÇÃO**

**WVW\_SetCodePage( nWinNum, nCodePage )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nCodePage**

Código de página.

## **DESCRIÇÃO**

Define a página de código da janela atual.

## **RETORNO**

.T. se conseguiu definir a nova página de código, .F. caso contrário.

## **FUNÇÃO**

**WVW\_SetDefCentreWindow( lCentre )**

## **PARÂMETROS**

**lCentre**

Se informado .T., configura para que todas as janelas posteriormente abertas, sejam centralizadas horizontalmente e verticalmente.

Se não informado, retorna o valor da configuração atual.

## **DESCRIÇÃO**

Lê ou atualiza o parâmetro de centralização horizontal/vertical das janelas.

## **RETORNO**

Se **lCentre** informado, retorna o valor anterior da configuração, caso contrário retorna a configuração atual.

## **FUNÇÃO**

**WVW\_SetDefHCentreWindow( lCentre )**

## **PARÂMETROS**

**lCentre**

Se informado .T., configura para que todas as janelas posteriormente abertas, sejam centralizadas horizontalmente.

Se não informado, retorna o valor da configuração atual.

## **DESCRIÇÃO**

Lê ou atualiza o parâmetro de centralização horizontal das janelas.

## **RETORNO**

Se **lCentre** informado, retorna o valor anterior da configuração, caso contrário retorna a configuração atual.

## **FUNÇÃO**

**WVW\_SetDefLineSpacing( nLineSpacing )**

## **PARÂMETROS**

**nLineSpacing**

Número do espaçamento entre linhas. Deve ser menor ou igual a 40, caso contrário será ignorado. Se não informado lê a configuração atual.

## **DESCRIÇÃO**

Seta o espaçamento entre linhas de todas as janelas.

## **RETORNO**

A configuração anterior.

## FUNÇÃO

**WVW\_SetDefLSPACECOLOR( nColorIndex )**

## PARÂMETROS

**nColorIndex**

O índice da cor para o espaçamento entre linhas. Deverá ser entre 0 e 15 ( os índices são os mesmos do Clipper ) ou -1, indicando que não tem cor.

## DESCRIÇÃO

Altera a cor para o espaçamento entre linhas.

## RETORNO

A configuração antiga, se **nColorIndex** for informado, caso contrário a configuração atual.

## **FUNÇÃO**

**WVW\_SetDefVCentreWindow( lCentre )**

## **PARÂMETROS**

**lCentre**

Se informado .T., configura para que todas as janelas posteriormente abertas, sejam centralizadas verticalmente.

Se não informado, retorna o valor da configuração atual.

## **DESCRIÇÃO**

Lê ou atualiza o parâmetro de centralização vertical das janelas.

## **RETORNO**

Se **lCentre** informado, retorna o valor anterior da configuração, caso contrário retorna a configuração atual.

## FUNÇÃO

**WVW\_SetFont( cFontFace, nFontHeight, nFontWidth, nFontWeight, nFontQuality )**

## PARÂMETROS

### **cFontName**

Nome da fonte ( por exemplo, MS Sans Serif ).

### **nFontHeight**

Altura da fonte.

### **nFontWidth**

Largura da fonte.

### **nFontWeight**

“Peso” da fonte.

### **nFontQuality**

Qualidade da fonte.

Todos os parâmetros são opcionais. A função usa os valores que são atribuídos na criação da janela.

## DESCRIÇÃO

Modifica a fonte ou os parâmetros ligados a fonte ( altura, largura, qualidade, etc. )

## RETORNO

.T. se conseguiu alterar a fonte ou seus parâmetros, .F. caso contrário.



## FUNÇÃO

**WVW\_SetIcon( nWinNum, nIcon, cIcon )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nIcon**

Identifica o ícone no arquivo de recursos.

**cIcon**

O nome do arquivo de ícone.

## DESCRIÇÃO

Configurar o ícone da janela atual ( o parâmetro **nWinNum** é ignorado ).

## RETORNO

.T. se conseguiu ler o ícone, .F. caso contrário.

## **FUNÇÃO**

**WVW\_SetLastMenuEvent( nWinNum, nMenuEvent )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nMenuEvent**

Item do menu.

## **DESCRIÇÃO**

Configura o último item selecionado do menu ( retorno da função WVW\_GetLastMenuEvent() ).

## **RETORNO**

Última configuração válida.

## FUNÇÃO

**WVW\_SetLineSpacing( nWinNum, nLineSpacing )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela corrente.

**nLineSpacing**

Se informado, altera o espaçamento entre linhas da janela **nWinNum**, caso contrário lê a configuração atual do parâmetro.

## DESCRIÇÃO

Lê/Configura o espaçamento da janela. Se o tamanho da janela ficar muito grande, o valor anterior será restaurado. Essa função atualiza apenas a janela corrente. Para configurar globalmente, utilize WVW\_SETDEFLINESPACING().

## RETORNO

A configuração anterior, se **nLineSpacing** for informado, ou a configuração atual.

## FUNÇÃO

**WVW\_SetLSpaceColor( nWinNum, nColorIndex )**

## PARÂMETROS

**nWinNum**

Número da janela atual. O default é a janela corrente.

**nColorIndex**

O índice da cor para o espaçamento entre linhas. Deverá ser entre 0 e 15 ( os índices são os mesmos do Clipper ) ou -1, indicando que não tem cor.

## DESCRIÇÃO

Altera a cor para o espaçamento entre linhas da janela **nWinNum**. Se desejar alterar a configuração de todas as janelas, use a função WVW\_SetDefLSpaceColor()

## RETORNO

A configuração antiga, se **nColorIndex** for informado, caso contrário a configuração atual.

## FUNÇÃO

WVW\_SetMainCoord( lMainCoord )

## PARÂMETROS

**lMainCoord**

Se informado, muda o modo de exibição das janelas. Caso contrário, lê o modo que está configurado.

## DESCRIÇÃO

Retorna ou o modo que está configurado o sistema ( Standard ou MainCoord ) ou, se o parâmetro **lMainCoord** for informado, troca a forma de configuração do modo de exibição das janelas.

## RETORNO

O modo antigo da configuração, se **lMainCoord** informado, ou o modo atual.

## FUNÇÃO

**WVW\_SetMaxBMCache ( nMaxCache )**

## PARÂMETROS

**nMaxCache**

Número máximo de bitmaps no cache. O default é 20 e o mínimo é 1.

## DESCRIÇÃO

Para minimizar a operação de carregar um bitmap, a função **WVW\_DrawImage()** faz um cache de bitmaps quando ele é lido do disco. Dessa forma, quando usamos **WVW\_DrawImage()** outras vezes, o bitmap já estará na memória.

Quando o cache de bitmaps chegar ao seu limite, sempre o último bitmap lido e carregado será descartado em favor do novo.

Não há nenhuma forma de descartar um bitmap específico do cache. Se achar conveniente, pode controlar o cache de bitmaps manualmente, através da função **WVW\_LoadPicture()**.

## RETORNO

Se o parâmetro **nMaxChace** não for informado, retorna a configuração atual do número máximo de bitmaps do cache, caso contrário, configura esse número.

## FUNÇÃO

**WVW\_SetMenu( nWinWin, hMenu )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**hMenu**

Contém as informações do menu.

## DESCRIÇÃO

Adiciona um menu, apontado por **hMenu**, a janela atual ( o parâmetro **nWinNum** é ignorado e a função procura a janela atual ).  
A janela então é redesenhada para ajustar seu tamanho.

## RETORNO

nenhum.

## **FUNÇÃO**

**WVW\_SetMenuKeyEvent( nWinNum, nKeyEvent )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nKeyEvent**

Número do evento definido pelo usuário para o comando do menu do windows.

## **DESCRIÇÃO**

Define um evento numérico do menu, para a janela atual.

## **RETORNO**

Última configuração válida.



## **FUNÇÃO**

**WVW\_SetMouseMove( nWinNum, lMouseMove )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**lMouseMove**

.T. reconhece o uso do mouse na aplicação.

.F. não reconhece o uso do mouse na aplicação.

Se o parâmetro for omitido, retorna a configuração atual.

## **DESCRIÇÃO**

Habilita/Desabilita o reconhecimento do mouse na aplicação.

## **RETORNO**

Se informado **lMouseMove**, retorna .T. ou .F., se conseguiu ou não configurar, caso contrário retorna a última configuração válida.

## FUNÇÃO

**WVW\_SetMousePos( nWinNum, nRow, nCol )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nRow, nCol**

Linha/Coluna para posicionamento do cursor do mouse.

## DESCRIÇÃO

Posiciona o mouse na posição indicada por **nRow/nCol**.

## RETORNO

.T. se conseguiu reposicionar o mouse, .F. caso contrário.

## **FUNÇÃO**

**WVW\_SetOnTop()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Deixa a janela sempre visível, mesmo que o foco esteja em outra janela. Esta função altera as propriedades da janela, de modo que ela sempre estará acima de qualquer outra janela aberta.

## **RETORNO**

.T. se conseguiu alterar o posicionamento da janela, .F. caso contrário.

## **FUNÇÃO**

**WVW\_SetPaintRefresh( nPaintRefresh )**

## **PARÂMETROS**

**nPaintRefresh**

Intervalo em milisegundos para chamar WVW\_PAINT(). Deve ser maior ou igual a 50 ou igual a 0, causando a execução imediata da repintura.

## **DESCRIÇÃO**

Determina o intervalo para a chamada da função WVW\_PAINT(), responsável pelo redesenho da tela.

## **RETORNO**

Valor antigo para **nPaintRefresh**.

## **FUNÇÃO**

**WVW\_SetPalette( aRGBValues )**

## **PARÂMETROS**

**aRGBValues**

Matriz com 16 elementos com valores RGB.

## **DESCRIÇÃO**

Configura a paleta de cores.

## **RETORNO**

nenhum.

## **FUNÇÃO**

**WVW\_SetPen( nPenStyle, nWidth, nColor )**

## **PARÂMETROS**

### **nPenStyle**

Define o estilo do traço. As definições estão no arquivo WINGDI.CH (PS\_SOLID, PS\_DOT, etc.).

### **nWidth**

Especifica a largura do traço.

### **nColor**

Índice da cor do traço.

## **DESCRIÇÃO**

Define o estilo, largura e cor do traço.

## **RETORNO**

.T. se conseguiu definir corretamente, caso contrário, .F.

## **FUNÇÃO**

**WVW\_SetPointer( nWinNum, nIcon )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nIcon**

Modelo do cursor a ser lido. Pode ser uma string que identifica um RESOUCE em um arquivo RC ou pode ser um modelo predefinido do windows ( o default é IDC\_ARROW ). Outras definições estão no arquivo WINUSER.CH )

## **DESCRIÇÃO**

Altera o cursor do mouse.

## **RETORNO**

Nenhum.

## FUNÇÃO

**WVW\_SetPopUpMenu**( **nWinNum**, **hPopUp** )

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela corrente.

**hPopUp**

Contém as informações do menu popup.

## DESCRIÇÃO

Configura o menu popup da janela atual ( o parâmetro **nWinNum** é ignorado, e a função procura a janela atual ).

## RETORNO

nenhum.



## FUNÇÃO

**WVW\_SetTimer( nWinNum, nInterval )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nInterval**

Intervalor em milisegundos.

## DESCRIÇÃO

Configura o evento **timer** para o intervalo de tempo **nInterval** ( útil para atualização de um relógio em uma status bar, por exemplo ).

Na prática essa função só irá funcionar se a função **WVW\_TIMER()** for definida no aplicativo.

## RETORNO

.T. se conseguiu configurar o **timer**, .F. caso contrário.

## **FUNÇÃO**

**WVW\_SetTitle( nWinNum, cTitle )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**cTitle**

Título da janela.

## **DESCRIÇÃO**

Define o título da janela atual ( o parâmetro **nWinNum** é ignorado ).

## **RETORNO**

nenhum.

## **FUNÇÃO**

**WVW\_SetTooltip( nWinNum, nTop, nLeft, nBottom, nRight, cToolText )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas para apresentação da tooltip.

**cToolText**

Texto para tooltip.

## **DESCRIÇÃO**

Define um tooltip.

## **RETORNO**

Nenhum.

## FUNÇÃO

**WVW\_SetTooltipActive( nWinNum, lToggle )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**lToggle**

Se for informado, habilita ( .T. ) ou desabilita ( .F. ) a exibição de tooltips.  
Se não for informado, retorna o estado atual ( se exibe ou não tooltips ).

## DESCRIÇÃO

Ativa/Desativa apresentação de tooltips.

## RETORNO

Configuração antiga ( se for informado o parâmetro **lToggle** ) ou configuração atual.

## FUNÇÃO

**WVW\_SetTooltipBkColor( nWinNum, nColor )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nColor**

Índice da cor para o tooltip.

## DESCRIÇÃO

Obtém ou altera a cor de fundo da tooltip.

## RETORNO

Se o parâmetro **nColor** não for informado, retorna a cor de fundo atual da tooltip, caso contrário, seta a nova cor de fundo.

## **FUNÇÃO**

**WVW\_SetTooltipMargin( nWinNum, nTop, nLeft, nBottom, nRight )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nTop, nLeft, nBottom, nRight**

Coordenadas para a tooltip.

## **DESCRIÇÃO**

Define as margens da tooltip. A margem é a distância ( em pixels ) entre as bordas da janela da tooltip e o texto contido dentro da tooltip.

## **RETORNO**

Nenhum.

## **FUNÇÃO**

**WVW\_SetTooltipText( nWinNum, cText )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**cText**

Texto para da tooltip.

## **DESCRIÇÃO**

Define um texto para a tooltip.

## **RETORNO**

Nenhum.

## FUNÇÃO

**WVW\_SetTooltipTextColor( nWinNum, nColor )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nColor**

Índice da cor para o texto da tooltip.

## DESCRIÇÃO

Define ou obtém a cor do texto da tooltip.

## RETORNO

Se o parâmetro **nColor** não for informado, retorna a cor atual do texto, caso contrário, seta a nova cor do texto para tooltip.



## **FUNÇÃO**

**WVW\_SetTooltipTitle( nWinNum, nIcon, cTitle )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nIcon**

Identifica se deve ser apresentado um ícone em conjunto com o texto da tooltip. É permitido um dos valores abaixo :

- 0**        nenhum ícone ( default )
- 1**        Informação
- 2**        Aviso
- 3**        Erro

Qualquer valor informado acima de **3**, o sistema converterá para **0**.

## **DESCRIÇÃO**

Define o título da tooltip.

## **DESCRIÇÃO**

Nenhum.

## FUNÇÃO

**WVW\_SetTooltipWidth( nWinNum, nWidth )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nWidth**

Tamanho para a área de tooltip.

## DESCRIÇÃO

Modifica ou obtém o tamanho da área de tooltip.

## RETORNO

Se o parâmetro **nWidth** não for passado, retorna o tamanho atual da área de tooltip, caso contrário, seta o tamanho da área de tooltip e retorna esse valor.

## **FUNÇÃO**

**WVW\_SetVertCaret( lOn )**

## **PARÂMETROS**

**lOn**

Se .T., muda a posição do cursor de texto para vertical.

Se .F., muda a posição do cursor de texto para posição horizontal.

Se não informado, lê a configuração atual.

## **DESCRIÇÃO**

Mudar a posição do cursor de texto.

## **RETORNO**

Se o parâmetro **lOn** for informado, retorna a configuração antiga, antes da mudança. Se não, retorna o valor atual da configuração.

## **FUNÇÃO**

**WVW\_SetWindowCentre( nWinNum, lCentre, lPaintIt )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **lCentre**

.T. centraliza a janela.

.F. não centraliza a janela.

### **lPaintIt**

.T. Redesenha de imediato as janelas.

.F. Apenas atualiza a configuração.

## **DESCRIÇÃO**

Atualiza o parâmetro para centralização de todas as janelas abertas ou que serão abertas. Se o parâmetro **lPaintIt** for .T. e lCentre também, todas as janelas são redesenhadas e centralizadas.

## **RETORNO**

nenhum.

## **FUNÇÃO**

**WVW\_SetWindowPos( nWinNum, nXPosition, nYPosition )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nXPosition**

Posição horizontal em pixels.

**nYPosition**

Posição vertical em pixels.

## **DESCRIÇÃO**

Muda a posição da janela atual, baseado nas coordenadas de **nXPosition** e **nYPosition**.

## **RETORNO**

nenhum.

## FUNÇÃO

**WVW\_SetWinStyle( nWinNum, nStyle )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela corrente.

**nStyle**

Novo estilo da janela. Se a janela tiver controle (PUSHBUTTON, SCROLLBAR, ETC. ), deve ser adicionado o estilo WS\_CLIPCHILDREN.

## DESCRIÇÃO

Lê ou seta o estilo da janela **nWinNum**. Se a janela estiver escondida ou minimizada, e se o parâmetro **nStyle** for informado, ela será apresentada.

## RETORNO

Retorna o estilo antigo da janela.

## **FUNÇÃO**

**WVW\_ShowWindow( nWinNum, nMode )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nMode**

Indica o modo como a janela será apresentada. O default é SW\_SHOWNORMAL ( as outras definições estão no arquivo WINUSER.CH ).

## **DESCRIÇÃO**

Configura o estado de apresentação da janela.

## **RETORNO**

**nenhum.**

## FUNÇÃO

**WVW\_TBAddButton( nWinNum, nCommand, xBitmap, cLabel, nBitmapType, lMap3DColors )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nCommand**

Identificador da ação que será executada quando o usuário pressionar o botão.

### **xBitmap**

Identificação do RESOURCE do bitmap em um arquivo RC ( nesse caso a imagem não pode ter mais que 256 cores ) ou caminho completo do arquivo da imagem.

### **cLabel**

Texto do botão. Se o parâmetro **lDisplayText** da função **WVW\_TBCreate()** estiver configurado para .T., o texto será apresentado logo abaixo da imagem, caso contrário será usado como tooltip.

### **nBitmapType**

Tipo do bitmap. Pode ter os seguintes valores :

- 0** Custom
- 1** Bitmaps standard do windows ( COPIAR, COLAR, PROCURAR, ABRIR, etc. )
- 2** Bitmaps "view" do windows ( são aqueles que mudam a visualização, como por exemplo, exibir detalhes, ordenar por nome, etc. )
- 3** Bitmap do Windows Explorer

### **lMap3DColors**

Se a imagem terá efeito de transparência (só tem sentido se o parâmetro **nBitmapType** for **0**).

## DESCRIÇÃO

Adiciona um botão em uma toolbar a direita dos botões já existentes.

## RETORNO

.T.



## **FUNÇÃO**

**WVW\_TBButtonCount()**

## **PARÂMETROS**

Nenhum.

## **DESCRIÇÃO**

Obtém o número de botões em uma toolbar.

## **RETORNO**

Número de botões em uma toolbar.

## FUNÇÃO

**WVW\_TBCMD2Index( nWinNum, nCmd )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nCmd**

Identificador do comando a pesquisar.

## DESCRIÇÃO

Obtém o número do botão que está associado o comando **nCmd**.

## RETORNO

Número do botão da toolbar que está associado ao comando **nCmd**, ou -1 se não conseguiu encontrar nenhuma associação.

## FUNÇÃO

**WVW\_TBCreate( nWinNum, lDisplayText, nStyle, nSystemBitmap, nImageWidth, nImageHeight )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **lDisplayText**

Se .T., o texto do botão da toolbar será mostrado juntamente com a imagem.

Se .F., o texto só aparecerá como tooltip ( default ).

### **nStyle**

Estilo da toolbar. O default é TBSTYLE\_FLAT | TBSTYLE\_TOOLTIPS ( outras definições podem ser encontradas no arquivo COMMCTRL.H ).

### **nSystemBitmap**

Indica se serão usados bitmaps do windows, e seus tamanhos. Os valores permitidos são :

- 0** Não usar bitmaps do sistema
- 1** Usar bitmaps do sistema de tamanho pequeno
- 2** usar bitmaps do sistema de tamanho grande

### **nImageWidth, nImageHeight**

Comprimento e largura da imagem ( só tem efeito se o parâmetro **nSystemBitmap** for **0** ).

## DESCRIÇÃO

Cria uma toolbar no topo da janela atual ( inicialmente sem nenhum botão ).

## RETORNO

Handle da toolbar, se operação for bem-sucedida, ou **0** caso contrário.

## **FUNÇÃO**

**WVW\_TBDeleteButton( nWinNum, nButton )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nButton**

Número do botão para excluir. O botão separador também conta como um botão e pode ser excluído.

ATENÇÃO : A lista de botões começa em zero, ou seja, **nButton** = 0 significa o botão mais a esquerda em uma toolbar.

## **DESCRIÇÃO**

Exclui um botão.

## **RETORNO**

.T. se conseguiu adicionar botão, .F. caso contrário.

## **FUNÇÃO**

**WVW\_TBDestroy()**

## **PARÂMETROS**

Nenhum.

## **DESCRIÇÃO**

Exclui uma toolbar da janela atual.

## **RETORNO**

Nenhum.

## **FUNÇÃO**

**WVW\_TBEnableButton( nWinNum. nButton, lToggle )**

## **PARÂMETROS**

### **nWinNum**

Número da janela. O default é a janela atual.

### **nButton**

Número do botão na toolbar ( a lista começa em zero ).

### **lToggle**

.T.   Habilita um botão  
.F.   Desabilita um botão

## **DESCRIÇÃO**

Habilita/Desabilita um botão em uma toolbar.

## **RETORNO**

.T. se conseguiu habilitar/desabilitar botão, .F. caso contrário.

## **FUNÇÃO**

**WVW\_TBIndex2CMD( nWinNum, nButton )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nButton**

Número do botão da toolbar ( começando com zero ).

## **DESCRIÇÃO**

Retorna o identificador de comando associado ao botão.

## **RETORNO**

Número do identificador de comando, se operação for bem-sucedida, ou -1 caso contrário.

## **FUNÇÃO**

**WVW\_TrackPopupMenu( nWinNum, nHandle )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nHandle**

Handle do menu popup.

## **DESCRIÇÃO**

Mostra um menu na posição do cursor.

## **RETORNO**

Identificador da opção selecionada ou **0** caso o usuário cancele ou retorne um erro.



## **FUNÇÃO**

**WVW\_UnreachedBr( nWinNum, nBottomPixels, nRightPixels )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela corrente.

**nBottomPixels**

Deve ser passado por referência, para guardar o número de linhas, em pixels.

**nRightPixels**

Deve ser passado por referência, para guardar o número de colunas, em pixels.

## **DESCRIÇÃO**

Ler o número de pixels que ficaram fora do alcance para linhas e colunas quando a janela **nWinNum** está maximizada.

## **RETORNO**

O número de linhas/colunas em pixels.

## **FUNÇÃO**

**WVW\_UpdateWindow()**

## **PARÂMETROS**

nenhum.

## **DESCRIÇÃO**

Atualiza a janela, redesenhando a tela.

## **RETORNO**

nenhum.

## FUNÇÃO

**WVW\_XBCreate( nWinNum, nStyle, nTop, nLeft, nLength, bBlock, aOffset )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nStyle**

0      Horizontal  
1      Vertical

**nTop, nLeft**

Linha/Coluna do ponto inicial da scrollbar ( em caracteres )

**nLength**

Comprimento da scrollbar( em caracteres )

**bBlock**

Bloco de código que será executado a cada evento VM\_VSCROLL/VM\_HSCROLL. Esse bloco de código será executado através dos seguintes parâmetros :

<b>nWinNum</b>	Número da janela
<b>nXBid</b>	Identificador da scrollbar
<b>nXBmsg</b>	Mensagem da scrollbar. Pode ser dividida em duas categorias :

As mensagens que precisam ser tratadas :

<b>SB_LINEUP/SB_LINELEFT</b>	0: botão para cima/para a esquerda pressionado
<b>SB_LINEDOWN/SB_LINERIGHT</b>	1: botão para baixa/para a direita pressionado
<b>SB_PAGEUP/SB_PAGELEFT</b>	2: página para cima/para esquerda
<b>SB_PAGEDOWN/SB_PAGERIGHT</b>	3: página para baixo/para direita

As mensagens de tratamento opcionais :

<b>SB_THUMBPOSITION</b>	4: O botão da scrollbar foi solto na posição nXBPos
<b>SB_THUMBTRACK</b>	5: O botão da scrollbar começou a ser arrastado na posição nXBPos
<b>SB_ENDSCROLL</b>	8: Indica o fim da rolagem da scrollbar

<b>nXBPos</b>	Posição do botão da scrollbar ( apenas se mensagem for SB_THUMBPOSITION ou SB_THUMBTRACK
---------------	--

**aOffset**

Matriz com quatro elementos contendo das coordenadas dos cantos, em pixels, para ajustar as dimensões da scrollbar.

## DESCRIÇÃO

Cria uma scrollbar para a janela atual.

## RETORNO

O handle da scrollbar, se a operação for bem-sucedida, caso contrário retorna **0**.

## FUNÇÃO

**WVW\_XBDestroy( nWinNum, nXBId )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nXBId**

Identificador da scrollbar.

## DESCRIÇÃO

Exclui a scrollbar **nXBId** da janela atual.

## RETORNO

Nenhum.

## **FUNÇÃO**

**WVW\_XBEnable( nWinNum, nXBId, nFlags )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nXBId**

Identificador da scrollbar.

**nFlags**

Os seguintes valores são permitidos :

- |          |                                      |
|----------|--------------------------------------|
| <b>0</b> | Habilita as duas setas               |
| <b>1</b> | Desabilita a seta para cima/esquerda |
| <b>2</b> | Desabilita a seta para baixo/direita |
| <b>3</b> | Desabilita as duas setas             |

## **DESCRIÇÃO**

Habilita/Desabilita uma scrollbar na janela atual.

## **RETORNO**

.T. se a operação foi realizada com sucesso, .F. caso contrário.

## FUNÇÃO

**WVW\_XBInfo( nWinNum, nXBId )**

## PARÂMETROS

**nWinNum**

Número da janela. O default é a janela atual.

**nXBId**

Identificador da scrollbar.

## DESCRIÇÃO

Obtém informações de uma scrollbar.

## RETORNO

Matriz com cinco elementos, contendo as seguintes informações :

<b>nMin</b>	Valor mínimo de posições para movimentação
<b>nMax</b>	Valor máximo de posições para movimentação
<b>nPageSize</b>	Tamanho da página
<b>nPos</b>	Posicionamento do botão de movimentação
<b>nTrackPos</b>	Posição exata do botão de movimentação quando o usuário começou uma operação de arrastar-e-soltar.

Essa matriz só retornará com esses valores caso a operação seja bem-sucedida. Se a operação solicitada for inválida, a função retornará uma matriz vazia.

## **FUNÇÃO**

**WVW\_XBShow( nWinNum, nXBId, lShow )**

## **PARÂMETROS**

**nWinNum**

Número da janela. O default é a janela atual.

**nXBId**

Identificador da scrollbar.

**lShow**

.T. apresenta a scrollbar ( default )

.F. esconde a scrollbar

## **DESCRIÇÃO**

Apresenta ou esconde uma scrollbar na janela atual.

## **RETORNO**

.T. se a operação foi realizada com sucesso, .F. caso contrário.



## FUNÇÃO

**WVW\_XBUpdate( nWinNum, nXBId, nPos, nPageSize, nMin, nMax )**

## PARÂMETROS

### **nWinNum**

Número da janela. O default é a janela atual.

### **nXBId**

Identificador da scrollbar.

### **nPos**

Especifica a posição do botão de movimentação da scrollbar.

### **nPageSize**

Tamanho da página. Esse valor é usado pela scrollbar para determinar o tamanho proporcional para o salto de página do botão de movimentação.

### **nMin**

Valor mínimo de posições para movimentação.

### **nMax**

Valor máximo de posições para movimentação.

## DESCRIÇÃO

Atualiza e reapresenta os dados de uma scrollbar.

Os parâmetros **nPos**, **nPageSize**, **nMin**, **nMax** são opcionais, entretanto, os parâmetros **nMin** e **nMax** ou devem ser informados os dois, ou nenhum deverá ser informado.

## RETORNO

A posição atual do botão de movimentação da scrollbar, se a operação for bem-sucedida, ou -1 caso contrário.

## FUNÇÃO

**WVW\_xReposWindow( lAnchored )**

## PARÂMETROS

**lAnchored**

Se **.T.** ( default ), todas as janelas são posicionadas de acordo com a sua respectiva coordenada ( row1, col1 ).

Se **.F.**, todas as janelas são posicionadas de acordo com parâmetro **CenterWindow** da função **WVW\_CENTERWINDOW()**.

## DESCRIÇÃO

Reposiciona todas as janelas para as suas posições iniciais. A janela principal será centralizada se não estiver maximizada.

## RETORNO

nenhum.