Table Of Contents

Introduction

Classes

- **WBColumn()**
- **WBrowse()**

Bitmaps

Vertical Scroll Bars

Navigation Keys

Icons

Technical Support

Introduction

WBrowse(T) is a TopClass version of WBrowse(s), a replacement for the tbrowse() class that is part of CA-Clipper.   It supports most of the instance variables and methods that tbrowse() does while adding some special ones for working in the Windows environment.   Class(y) and HighClass users please see the Technical Support section for information on ordering WBrowse(s).

WBrowse() Class.

Properties:

aAltKeys
alias
autolite
autoSize

beeper
boxCursor
buttonBlock

cargo
colCount
colFont
colGrids
colPos
colorSpec

destroyBlock
doubleclick

escape

findBlock

goBottomBlock
goTopBlock

headFont
headHeight
hitBottom
hitTop
hWnd

keyBlock

leftVisible
lineHeight

make3D
mimicButton
moveCursor
msgBlock

nCrecNo
nCurRec
nMaxLines
nMaxRec

nubAlign
nubBlock
nubHBitmap
nubHBitAlign

nubClickBlock
nubColor
nubHeading
nubHeadBlock
nubHeadColor
nubWidth

rightVisible
rowCount
rowGrids
rowPos

saveColSize
scanBlock
selectblock
showFocus
showNumbers
sizeCursor
skipBlock

tbAlign
tbBitMap
tbBitAlign
tbFont
tbHeight
tbText
tbTextColor

userMove
userSize

Methods:

addColumn()

bottom()

configure()

deHiLite()
delColumn()
destroy()
down()

end()

getCellData()
getCellRect()
getColumn()
getColCargo()
goBottom()
goTop()

hiLite()
highLiteCol()
highLiteRow()
home()

init()
insColumn()

left

modColumn()

panEnd()
panHome()
panLeft()
panRight()
pageDown()
pageUp()

refreshAll()
refreshCol()
refreshCurrent()
right()

setColBitMap()
setColCargo()
setColFont()
setColPicture()
setColWidth()
setHeadBlock()
setHeadColor()
setHeadHeight()
setRowHeight()
showData()
skip()

top()

up()

WBrowse:aAltKeys (Access/Assign)

This is an optional array containing scan codes and codeblocks to be evaluated when the user presses the Alt key and the key whose scan code is matched.   Using this approach you can establish a set of "hot keys/accelerators" to perform any action desired.

Example:

```
oB:aAltKeys := { {VK_A,{||AddRecord()}}, {VK_D,{||DeleteRecord()}} }
```

In the above example when the user presses Alt+A the AddRecord() function/procedure will be called, and pressing Alt+D calls the DeleteRecord() function/procedure.

WBrowse:alias (Access/Assign)

This variable is used by WBrowse() to select the correct alias/workarea to use when record/cursor movement or data refresh messages are received.   Please note that you **MUST** set this variable if you want WBrowse() to automatically select the correct database.

If you are browsing an array or using a database object then you **MUST** set alias to NIL.

Example:

```
oB:alias := "CHILDREN"  // Browsing CHILDREN.DBF
```

WBrowse:autoLite (Access/Assign)

Contains a logical value.   When autoLite is set to .T. the browse uses a highlight bar that highlights all visible columns.   When set to .F. the browse uses a highlight bar that highlights a single cell only.

Example:

```
oB:autoLite := .F.  // Highlight each cell
```

WBrowse:autoSize (Access/Assign)

This variable determines if WBrowse() will or will not automatically set internal properties based on fonts selected, width of data returned from the column blocks, window size, etc.

The default is .T.   If you set autoLite to .F. then it is up to you to set column widths, heading height, etc.

To manually set column widths and let WBrowse() handle the rest then set saveColSize to .T.

Example:

```
oB:autoSize := .F.  // We will set things ourself
```

See Also: **saveColSize**

WBrowse:beeper (Access/Assign)

This variable controls whether or not WBrowse() will beep when the user tries to move the cursor past the top, bottom, left or right most column/row. The default is on (.T.). To turn the beep off you set this variable to .F.

Example:

```
oB:beeper := .F. // Don't annoy the user
```

WBrowse:boxcursor (Access/Assign)

When set to .T. WBrowse() will use a cursor style that consists of a black box/rectangle.   Defaults to .F.

Example:

```
oB:boxcursor := .T. // Make the browse look like Excel
```

WBrowse:cargo (Access/Assign)

Contains a value of any type that can be retrieved later.

Example:

```
oB:cargo := oDB   // Carry our database object around with us
```

WBrowse:colCount (Access)

Contains a numeric value indicating the number of columns in the browse.

Example:

```
nCols := oB:colCount  // Keep track of columns in browse
```

WBrowse:colFont (Access/Assign)

This is a handle to a valid font that WBrowse() will use to display the data in the browse columns.   You can specify a different font for a specific column by using setColFont().

Please note that WBrowse() does not destroy this font automatically.

Example:

```
oB:colFont := hFont  // Set default data font
```

See Also: **setColFont()**

WBrowse:colGrids (Access/Assign)

When set to .F. WBrowse() will NOT draw grid lines between each browse column.   The default is .T.

Example:

```
oB:colGrids := .F.  // No vertical grids please
```

WBrowse:colPos (Access/Assign)

Contains a numeric value idicating the data column where the browse cursor is currently located.   Only valid if autoLite is .F..   If set directly refreshAll() **MUST** be sent to update the browse display.

Example:

```
oB:colPos := 3     // Move to DOB column
oB:refreshAll()    // Update the display
```

WBrowse:colorSpec (Access/Assign)

Unlike tbrowse() the colorSpec used by WBrowse() is an array with the following format:

{RGB() Value for un-selected foreground color, RGB() value for the un-selected background color, RGB() value for the selected foreground color, RBG() value for the selected background color}

Defaults to:

```
       {GetSysColor( COLOR_WINDOWTEXT ),;
        GetSysColor( COLOR_WINDOW ),;
        GetSysColor( COLOR_HIGHLIGHTTEXT ),;
        GetSysColor( COLOR_HIGHLIGHT )}
```

Example:

```
oB:colorSpec :=  RGB(0,0,0),   RGB(255,255,255),;  // Unselected
                 RGB(255,255), RGB(0,0,0)          // Selected
```

NOTE: If you specify a foreground and background color in your column definition then those colors will override the colorspec colors for the un-selected colors for that column.

WBrowse:destroyBlock (Access/Assign)

Codeblock to be evaluated when the browse is destroyed.   Can be used to destroy other browses that are linked to the browse being destroyed or to close databases, etc.

Example:

```
oB:destroyBlock := {|oB|DBCLOSEAREA( "CHILD" )}  // Close database on exit
```

WBrowse:doubleClick (Access/Assign)

This is a code block that is called each time the user double clicks on a browse row/cell. WBrowse() passes Self, nMsg, nWparam, and nLparam to the codeblock when it is evaluated.

Example:

```
oB:doubleClick := {|oB,nMsg,nWparam,nLparam|EditRec( oB )} // Do edit routine
```

WBrowse:escape (Access/Assign)

When set to .T. (default) pressing the Esc key will terminate the browse.

Example:

```
oB:escape := .F. // No messy exits please...
```

WBrowse:findBlock (Access/Assign)

Code block for searching.   Called when the user presses the F2 key.

Example:

```
oB:findBlock := {|oB|FindRec( oB )} // Press f2 to search...
```

WBrowse:goBottomBlock (Access/Assign)

Contains a code block that is executed in response to the goBottom() message.   Defaults to GO BOTTOM.   A user supplied block must not only position the record pointer on the last record to be displayed but be responsible for setting the following instance variables:

```
      hitBottom - set to logical .T.
      hitTop    - set to logical .F.
      rowPos    - set to rowCount
      nCurRec   - set to LASTREC()
      nCrecNo   - set to RECNO()
      nMaxRec   - set to LASTREC()
```

Example:

The following is the default bock definition used by WBrowse():

```
STATIC PROCEDURE BGoBottom( oB )

      DBGOBOTTOM()
      oB:hitTop     := BOF()
      oB:hitBottom  := .T.
      oB:rowPos     := oB:nMaxLines
      oB:nCurRec    := LASTREC()
      oB:nCrecNo    := RECNO()
      oB:nMaxRec    := LASTREC()
RETURN
```

See Also: **bottom()**

WBrowse:goTopBlock (Access/Assign)

Contains a code block that positions the database to the first record to be browsed. Defaults to GO TOP.   A user supplied block is responsible for setting the following instance variables:

```
      hitBottom - set to logical .F.
      hitTop    - set to logical .T.
      rowPos    - set to 1
      nCurRec   - set to 1
      nCrecNo   - set to RECNO()
```

Example:

The following is the default block definition used by WBrowse():

```
STATIC PROCEDURE BGoTop( oB )

       DBGOTOP()
       oB:hitTop    := .T.
       oB:hitBottom := EOF()
       oB:rowPos    := 1
       oB:nCurRec   := 1
       oB:nCrecNo   := RECNO()
       oB:nMaxRec   := LASTREC()
RETURN
```

See Also: **top()**

WBrowse:headFont (Access/Assign)

This must be a handle to a valid font.   WBrowse() uses this font to display text on the column headings and nubs.

Example:

```
oB:headFont := hBFont // Use bigger font for headings...
```

WBrowse:headHeight (Access/Assign)

This variable is used to manually control the height of the column headings.   The value is in pixels.

Example:

```
oB:autoSize   := .F. // Tell WBrowse we will set things...
oB:headHeight := 24  // Make column headings 24 pixels high
```

WBrowse:hitBottom (Access/Assign)

Set to logical .T. to indicate browse is positioned on last record to be displayed.   You should set hitBottom to .F. before calling goBottom().

Example:

```
oB:hitBottom := .F. // Override current setting...
oB:goBottom()       // Go to last record
```

WBrowse:hitTop (Access/Assign)

Set to logical .T. to indicate browse is positioned on first record to be displayed.   You should set hitTop to .F. before calling goTop().

Example:

```
oB:hitTop := .F. // Override current setting...
oB:goTop()       // Go to first recordd
```

WBrowse:hWnd (Access)

Window handle of browse window.   You can use this window handle to send Windows messages directly to the browse.

Example:

```
SendMessage( oB:hWnd, WM_SYSCOMMAND, SC_CLOSE, 0 ) // Close browse
```

WBrowse:keyBlock (Access/Assign)

Code block to be evaluated when WM_KEYDOWN message is received.

Can be used for "in cell editing" routines, etc.   You **MUST** return .T. from your code block to tell WBrowse() that you processed the key or WBrowse() will handle it normally.

When called WBrowse() passes Self, nMsg, nWparam, and nLparam.

Example:

```
oB:keyBlock := {||oB,nMsg,nWparam,nLparam|;
               CellEdit( oB, nMsg, nWparam, nLparam )} // Cell edit
```

WBrowse:leftVisible (Access)

Contains a numeric value indicating the left most column currently displayed.

Example:

```
nLeftCol := oB:leftVisible // Last column on screen?
```

WBrowse:lineHeight (Access/Assign)

This variable is used to manually control the height of the browse rows.   The value is in pixels.

Example:

```
oB:autoSize   := .F. // Tell WBrowse we will set things
oB:lineHeight := 20  // Make line height 20 pixels
```

WBrowse:mimicButton (Access/Assign)

This variable is used to control whether or not the column headings and "nubs" will mimic the action of a pushbutton when clicked on.

The default is .F.

Example:

```
oB:mimicButton := .T. // Our headings will work like buttons
```

WBrowse:moveCursor (Access/Assign)

This is a handle to a valid cursor.   If set, WBrowse() will use the specified cursor when the user is moving/dragging a column.

Example:

```
hMoveCur      := LoadCursor( _GetInstance(), ID_MOVECUR )
oB:moveCursor := hMoveCur
```

WBrowse:msgBlock (Access/Assign)

Code block to be evaluated when the browse window receives a message.

Can be used for special routines.   You **MUST** return .T. from your code block to tell WBrowse() that you processed the message or WBrowse() will handle it normally.

When called WBrowse() passes Self, nMsg, nWparam, and nLparam.

Example:

```
oB:msgBlock := {|oB,nMsg,nWparam,nLparam|;
               SyncChild( oB, nMsg, nWparam, nLparam )} // Keep child in
sync...

// This function sends keystrokes to a child browse so they move
// in sync with each other...

STATIC FUNCTION SyncChild( oB, nMsg, nWparam, nLparam )

      IF ( nMsg == WM_KEYDOWN )
         SendMessage( oB2:hWnd, nMsg, nWparam, nLparam )
      ENDIF
RETURN( .F. ) // Make sure we let main browse handle this message
```

WBrowse:nCrecNo (Access/Assign)

Contains the value of the current physical record number.

Example:

```
nRec := oB:nCrecNo // Find out what record we are currently on...
```

WBrowse:nCurRec (Access/Assign)

This variable contains a numeric value indicating the current logical record.   It is used to calculate the position for the vertical thumb.

Example:

```
nLogRec := oB:nCurRec // Find logical record position
```

WBrowse:nMaxLines (Access)

Maximum number of browse rows that will fit within the client area of the browse window.

Example:

```
nMax := oB:nMaxLines // How many visible rows in browse window?
```

WBrowse:nMaxRec (Access/Assign)

This variable contains a numeric value indicating the current number of records in the database/array being browsed.   The user is responsible for updating this value each time a record is added or deleted.

Example:

```
oDB:DBDELETE()                  // Delete current record
oB:nMaxRec := oDB:LASTREC() - 1 // Update record count
oB:refreshAll()                 // Update browse...
```

WBrowse:nubAlign (Access/Assign)

Controls the way text or bitmap on the "nubs" are aligned.   Valid values are DT_LEFT, DT_CENTER, and DT_RIGHT.

Defaults to DT_CENTER.

Example:

```
oB:nubAlign := DT_LEFT // Left align text on nubs
```

WBrowse:nubBlock (Access/Assign)

This variable can be used to display data other than the record number on the "nubs". MUST be a codeblock.   Evaluated for each record read.

Example:

```
oB:nubBlock := {||IIF( CHILD->SEX == 'M', "Male", "Female" ) } // Show sex on
nubs
```

WBrowse:nubHBitmap (Access/Assign)

Used to display a bitmap on the nub heading.   Must be a handle to a bitmap in your resource file or one of the standard Windows bitmaps.

Example:

```
oB:nubHBitmap := OBM_CHECK  // Use Windows check mark bitmap on nub heading
```

WBrowse:nubHBitAlign (Access/Assign)

Controls the way the bitmap on the "nub" heading is aligned.   Valid values are DT_LEFT, DT_CENTER, and DT_RIGHT.

Defaults to DT_CENTER.

Example:

```
oB:nubHBitAlign := DT_RIGHT  // Make room for text also...
```

WBrowse:nubClickBlock (Access/Assign)

This codeblock will be evaluated if the user clicks on a "nub". The record pointer and browse cursor are first positioned on the line where the "nub" is located.

The current value of oB:rowPos can be used to determine which "nub" the user clicked on.

Example:

```
oB:nubClickBlock := {|oB|ViewRecord( oB ) } // Click on nub to view entire
record
```

WBrowse:nubColor (Access/Assign)

A numeric value of the type returned by RGB() which is used to set the textcolor for the "nubs".

Defaults to RGB(0,0,0).

Example:

```
oB:nubColor := RGB(0,0,128)  // Make nub text color light blue
```

WBrowse:nubHeading (Access/Assign)

A character constant or expression that is to be displayed on the "nub" column heading.

To display more than one line in the heading you pass an array containing each line.

Example:

```
oB:nubHeading := "Sex" // Set nub heading...
```

WBrowse:nubHeadBlock (Access/Assign)

A codeblock to be evaluated when the user clicks on the "nub" column heading.   If set the "nub" column heading acts like a button when clicked.

Example:

```
oB:nubHeadBlock := {|oB|SetNubs( oB ) } // Let user select nub display
```

WBrowse:nubHeadColor (Access/Assign)

A numeric value of the type returned by RGB() which is used to set the textcolor for the "nub" heading.

Defaults to RGB(0,0,0).

Example:

```
oB:nubHeadColor := RGB(255,255,255)  // Use white on gray for nub heading...
```

WBrowse:nubWidth (Access/Assign)

Used to manually set the width of the "nubs".   Value is in pixels.

Example:

```
oB:showNumbers := .T. // Turn on nubs
oB:nubWidth    := 40  // Set nub width to 40 pixels
```

See Also: **setColWidth()**

WBrowse:buttonBlock (Access/Assign)

Code block to be evaluated when mouse button messages are received.   Can be used to display a dialog for the user to do something with a column, etc.

When called WBrowse() passes Self, nMsg, nWparam, and nLparam.

Example:

```
// Let user change things with the right mouse button

oB:buttonBlock := {|oB,nMsg,nWparam,nLparam|
                   ModBrowse( oB, nMsg, nWparam, nLparam )}
```

WBrowse:rightVisible (Access)

Contains the column number of the right most column currently visible.

Example:

```
nRightCol := oB:rightVisible  // Right most visible column in browse...
```

WBrowse:rowCount (Access)

Contains a numeric value indicating the number of rows visible in the browse window.

Example:

```
nRows := oB:rowCount // How many visible rows in browse?
```

WBrowse:rowGrids (Access/Assign)

When set to .T. (default), WBrowse() will draw lines between each browse row.

Example:

```
oB:rowGrids := .F. // We don't want lines in our listbox browse...
```

WBrowse:rowPos (Access/Assign)

Contains a numeric value indicating the current row where the browse cursor is positioned. If set directly then refreshAll() **MUST** be used to update the cursor.

Example:

```
oB:rowPos := oB:rowPos + 1 // Move cursor down one row
oB:refreshAll()            // Update display
```

WBrowse:saveColSize (Access/Assign)

Setting this variable to .T. will cause configure() to keep the currently set column widths

Defaults to .F.

Example:

```
oB:autoSize := .T.         // Let WBrowse set things
oB:setColWidth( 1, 40 )    //
oB:setColWidth( 2, 100 )   //
oB:setColWidth( 3, 80 )    //
oB:saveColSize := .T.      // Except for our column widths!
```

WBrowse:scanBlock (Access/Assign)

This code block is used for first letter searching.   You must supply a function that locates a record based on the key the user presses.

Example:

The following is the default/internal scanBlock used by WBrowse():

```
STATIC PROCEDURE FirstKey( oB, cKey )
      LOCAL nCrec

      IF oB:nCrecNo == NIL
         RETURN
      ENDIF
      IF INDEXORD() != 0 .AND. RIGHT(UPPER(ORDBAGEXT()),3) == "NTX"
         nCrec := RECNO()
         SEEK cKey
         IF !FOUND()
            GOTO nCrec
            IIF(oB:beeper,MessageBeep( MB_ICONEXCLAMATION ),'')
         ELSE
            oB:nCurRec   := NtxPos( INDEXORD(), RECNO() )
            oB:nCrecNo   := RECNO()
            oB:rowpos    := 1
            oB:showData()
         ENDIF
      ENDIF
RETURN
```

WBrowse:selectBlock (Access/Assign)

This is a code block that is called each time the browse cursor is re-positioned.   The browse object, the current logical record number (nCurRec), and .T. or .F. (depending on whether or not the browse window has focus) is automatically passed to the code block.

Example:

```
oB:selectBlock := {|oB,nRecNo,lFocus|UpDateCtls(oB,nRecNo,lFocus)} // Update
controls
```

WBrowse:make3D (Access/Assign)

If set to .T. WBrowse() will draw the browse so column/row data appears to be raised/recessed.   Since there is a lot of overhead involved in drawing the browse data in this mode you should use it only with small browse windows.

Defaults to .F.

Example:

```
oB:make3D := .T. // Make browse 3D...
```

WBrowse:showFocus (Access/Assign)

If set to .T. WBrowse() will draw a focus rectangle around the currently selected cell. Defaults to .F.

Example:

```
oB:showFocus := .T. // Mimic Windows focus rectangle
```

WBrowse:showNumbers (Access/Assign)

This variable is used to control the display of information on the "nubs" at the start of each browse line.   If you use this feature you **MUST** set the size of the "nubs" manually.

If you do not specify a codeblock for nubBlock then WBrowse() will display the value of nCurRec on the "nubs".

Example:

```
oB:showNumbers := .T.  // Show logical record number on nubs
oB:nubWidth    := 40   // Make nubs 40 pixels wide...
```

WBrowse:sizeCursor (Access/Assign)

A handle to a valid cursor that is to be displayed when the user is re-sizing a column.

Example:

```
// Use our own cursor when user is re-sizing a column...

hSizeCur      := LoadCursor( _GetInstance(), ID_SIZECUR )
oB:sizeCursor := hSizeCur
```

WBrowse:skipBlock (Access/Assign)

Contains a code block that repositions the database.   This block must return the number of records actually skipped.   If the value returned is not the same as the number requested then the browse assumes that the operation encountered EOF() or BOF(). A user assigned block must also set the following:

```
     nCurRec - set to current logical record
     nCrecNo - set to RECNO()
```

Example:

The following is the default block definition used by WBrowse():

```
STATIC FUNCTION Skipper( oB, nSkip )

       DBSKIP( nSkip )
       DO CASE
          CASE BOF()
               oB:nCurRec    := 1
               oB:nCrecNo    := RECNO()
               oB:hitTop     := .T.
               oB:hitBottom  := EOF()
          CASE EOF()
               oB:nCurRec    := RECNO()
               oB:nCrecNo    := RECNO()-1
               oB:hitBottom  := .T.
               oB:hitTop     := BOF()
          OTHERWISE
               oB:nCurRec    += nSkip
               oB:nCrecNo    := RECNO()
               oB:hitTop     := .F.
               oB:hitBottom  := .F.
               RETURN( nSkip )
       ENDCASE
RETURN( 0 )
```

NOTE: When writing your own skip block please keep in mind that WBrowse() expects the record pointer to be positioned at LASTREC() + 1 when EOF() is encountered.

See Also: **skip()**

WBrowse:tbAlign (Access/Assign)

Title bar text alignment.   Must be DT_LEFT, DT_CENTER, or DT_RIGHT.

Defaults to DT_CENTER.

Example:

```
oB:tbAlign := DT_LEFT // Left align title bar text
```

WBrowse:tbBitMap (Access/Assign)

A valid handle to a bitmap resource to be displayed on the browse title bar.

Example:

```
// Load and set our title bar bitmap

hBitMap     := LoadBitmap( _GetInstance(), ID_TBBIT )
oB:tbBitMap := hBitMap
```

WBrowse:tbBitAlign (Access/Assign)

Title bar bitmap alignment.   Must be DT_LEFT, DT_CENTER, or DT_RIGHT.

Defaults to DT_CENTER.

Example:

```
oB:tbBitAlign := DT_RIGHT // Put our title bar bitmap on the right side...
```

WBrowse:tbFont (Access/Assign)

Must be a handle to a valid font to be used to draw the text on the browse title bar.

Example:

```
// Use different font for title bar...

hTFont    := CreateFont( aTFont )
oB:tbFont := hTFont
```

WBrowse:tbHeight (Access/Assign)

A value in pixels to be used when drawing the browse title bar.

Example:

```
oB:tbHeight := GetTextHeight( oB:hWnd, oB:tbFont ) // Set title bar height..
```

WBrowse:tbText (Access/Assign)

A string value to be displayed on the browse title bar.

Example:

```
oB:tbText := "Children Currently Enrolled" // Title bar text...
```

WBrowse:tbTextColor (Access/Assign)

A valid RGB() value to be used for the text on the browse title bar.

Example:

```
oB:tbTextColor := RGB( 0,128, 0 ) // Set title bar text to green on gray
```

WBrowse:userMove (Access/Assign)

If set to .T. (default), the user can move the visible columns by clicking on the column heading and "dragging" the column to it's new position.   If set to .F. this feature is disabled.

Example:

```
// Since we are going to let the user move columns around we
// need to set our column number in the column cargo slot
// so we can keep track of what column is where...

oB:setColCargo( 1, 1 )
oB:setColCargo( 2, 2 )
oB:setColCargo( 3, 3 )
oB:userMove := .T.
```

WBrowse:userSize (Access/Assign)

This variable determies if the user can re-size columns or not.   The default is .T..   To prevent the user from changing the size of the browse columns you must set this to .F..

If set to .T. the user can re-size the columns by clicking on the space between the column headings and then "dragging" the line to the left or right.   The minimum column width is 20 pixels and WBrowse() will automatically adjust the column width to 20 if the user tries to make it smaller.

Example:

```
oB:userSize := .F. // Don't let the user muck up the browse!
```

WBrowse:addColumn() Method
Add a column to a browse.

Syntax:

addColumn(<columnObject>) -> **Self**

Arguments:

<columnObject> is a valid column object returned from WBColumn

Returns:

Self.

Description:

Adds a new WBColumn object to the WBrowse object.   The column is added to the right and colCount is increased by one.

You should call configure() after adding a column.   If you are adding mutiple columns you only need to call configure() once after all columns have been added.   If you are adding columns to a newly created browse that is not yet visible then you do not need to call configure() since it will be automatically called when you make the browse visible by calling either goTop() or goBottom() for the first time.

Example:

```
oCol         := WBColumn
oCol:heading := "Name"
oCol:block   := ||CHILD->NAME
oCol:align   := DT_LEFT
oB:addColumn( oCol )
```

WBrowse:bottom()

Move to bottom of database/array.

Syntax:

bottom() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Moves the browse record pointer to the bottom of the database/arry being browsed.   This method simply evaluates the goBottomBlock.

Example:

```
oB:bottom()  // Move to bottom of file...
```

WBrowse:configure()Method

Configure browse.

Syntax:

configure() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Causes the WBrowse() object to update key instance variables and to re-calculate column widths.   Should be called anytime you modify any key browse property.

Example:

```
oB:setColPicture( 1, "99,999.99" )
oB:setColWidth( 1, 60 )
oB:configure()
oB:refreshAll()
```

WBrowse:deHilite()Method

De-highlight current row

Syntax:

deHilite() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Turns the browse cursor off.

Example:

```
oB:deHilite()  // Turn browse cursor off...
```

WBrowse:delColumn()Method
Delete a browse column.

Syntax:

delColumn(<nPos>) -> **objColumn**

Arguments:

<nPos> position of column to delete.   Columns are numbered from left to right starting with 1.

Returns:

Column object that is deleted if successful, NIL if an invalid column number is passed.

Description:

Deletes the WBColumn object at <nPos>.   The WBColumn object is returned so it can be re-inserted/re-added later.

Example:

```
oCol := oB:delColumn( 1 ) // Delete 1st column..
oB:addColumn( oCol )      // and add it back on the right
oB:configure()
oB:refreshAll()
```

WBrowse:destroy()Method

Destroy the browse.

Syntax:

destroy() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Destroys the browse and frees it's associated resources.   Should be called when the browse is to be terminated.

NOTE: This method is automatically called when the browse window is closed or sent a WM_DESTROY message by Windows.

Example:

```
oB:destroy()  // Kill the browse...
```

WBrowse:down()Method

Move browse cursor down one row.

Syntax:

down() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Moves the browse cursor down one row.   If the cursor is already positioned on the bottom row then the display is scrolled up.

Example:

```
oB:down()  // Move to next row...
```

WBrowse:end()Method

Move to rightmost column.

Syntax:

end() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Moves the browse cursor to the rightmost data column currently visible.   Not applicable when autoLite is set to .T.

Example:

```
oB:end()  // Move to right most column...
```

WBrowse:getCellData()Method

Retrieve current cell data.

Syntax:

getCellData() -> **{data,picture}**

Arguments:

None.

Returns:

An array containing the cell data and the picture clause for the selected column.

Description:

This method returns an array containing the contents of the currently highlighted cell and the picture clause if any. Only valid if autoLite is .F.

Example:

```
EditCell( oB:getCellData() ) // Edit cell contents...
```

WBrowse:getCellRect()

Get coordinates and size of current cell.

Syntax:

getCellRect() -> **{nLeft,nTop,nWidth,nHeight}**

Arguments:

None.

Returns:

An array containing the left, top, width and height of the currently selected cell.

Description:

This method returns an array containing the screen coordinates and size of the currently highlighted cell. Only valid if autoLite is .F. **Example:**

```
aRect := oB:getCellRect() // Get cell location/size...
```

WBrowse:getColumn()Method
Retrieve a browse column object.

Syntax:

getColumn(<nCol>) -> **objColumn**

Arguments:

<nCol> is the column number to retrieve.   Columns are numbered from left to right and start with 1.

Returns:

Column object for specified column or NIL if <nCol> is invalid.

Description:

Returns the WBColumn object specified by <nCol>.

Example:

```
// Retrieve and save browse column configuration...

FOR nCtr := 1 TO oB:colCount
   oCol := oB:getColumn( nCtr )
   AADD( aCols, oCol )
NEXT
```

WBrowse:getColCargo()Method
Retrieve column cargo.

Syntax:

getColCargo(<nCol>) -> **xCargo**

Arguments:

<nCol> is the column to retrieve the cargo from.

Returns:

The data in the specified columns cargo slot or NIL if an invalid column number is passed.

Description:

getColCargo() is used to get the cargo for a columns cargo slot.

Example:

```
// Find and delete a column...
STATIC FUNCTION DeleteCol( oB, nCol )

      FOR nCtr := 1 TO oB:colCount
          IF ( oB:getColCargo( nCtr ) == nCol )
             RETURN( oB:delColumn( nCtr ) )
          ENDIF
      NEXT
RETURN( NIL )
```

See Also: **setColCargo()**

WBrowse:goBottom()Method

Move browse cursor to bottom of file/array.

Syntax:

goBottom() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Repositions the database to the logical end of the file.

Example:

```
oB:hitBottom := .F. // Just in case we are already there
oB:goBottom()       // Move to end of file...
```

WBrowse:goTop()Method

Move browse cursor to top of file/array

Syntax:

goTop() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Repositions the database to the logical beginning of the file.

Example:

```
oB:hitTop := .F // Just in case we are already there...
oB:goTop()      // Move to top of file/array...
```

WBrowse:hiLite()Method

Turn browse cursor on.

Syntax:

hilite() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Turns the browse cursor on after it has been turned off using deHiLite().

Example:

```
oB:hiLite()
```

WBrowse:highLiteCol()Method
Highlight an entire column.

Syntax:

highLiteCol(<nCol>) -> **Self**

Arguments:

<nCol> is the number of the browse column to highlight.

Returns:

Self.

Description:

highLiteCol() is used to highlight a browse column.

The column specified by < nCol > is highlighted using the color specified in colorSpec for selected text.   The column remains highlighted until the next screen refresh.

Example:

```
// User is modifying a column so we highlite it...

oB:highLiteCol( nCol )
ModColumn( oB, nCol )
```

WBrowse:highLiteRow()Method

Highlight a browse row.

Syntax:

highLiteRow(<nRow>) -> **Self**

Arguments:

<nRow> is the visible row number to highlight.

Returns:

Self.

Description:

highLiteRow() is used to highlight a browse row.

The row specified by < nRow > is highlighted using the color specified in colorSpec for selected text.   The row remains highlighted until the next screen refresh.

Example:

```
oB:highLiteRow( 4 ) // Highlight browse row 4...
```

WBrowse:home()Method

Move to leftmost column.

Syntax:

home() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Repositions the browse cursor on the leftmost visible column. Not applicable when autoLite is set to .T.

Example:

```
oB:home() // Move to first column...
```

WBrowse:Init()Method

Construct a new WBrowse object.

Syntax:

WBrowse{ <hWnd>, <nLeft>, <nTop>, <nWidth>, <nHeight>,
[<cTitle>], [<nStyle>] } -> ***objWBrowse***

Arguments:

<hWnd> - Handle to parent window

<nLeft> - Column number (in pixels) of top left corner of browse window

<nTop> - Row number (in pixels) of top left corner of browse window

<nWidth> - Width in pixels of browse window

<nHeight> - Height in pixels of browse window

[<cTitle>] - Optional title for browse window

[<Style>] - Optional style for browse window. Must be a valid combination of WS_ values used in the CreateWindow() function call. defaults to WS_CAPTION+WS_SYSMENU+WS_MINIMIZEBOX+WS_THICKFRAME+ WS_MAXIMIZEBOX+WS_VSCROLL+WS_HSCROLL.

Returns:

A new WBrowse object.

Description:

Returns a new WBrowse object within a window specified by the cordinates provided. The WBrowse object is created with no columns and the default code blocks for data source positioning. The default code blocks execute the GO TOP, GO BOTTOM, and SKIP operations. Since there are no columns you must add columns for each column to be displayed.

Example:

```
// Create new browse...
oB := WBrowse{ hMainWnd, 100, 100, 500, 400, "Child List" }
```

WBrowse:insColumn()Method
Insert a new browse column.

Syntax:

insColumn(<nPos>, <oCol>) -> **oCol**

Arguments:

<nPos> is the position to insert <oCol> which is a valid WBColumn object .

Returns:

<oCol> if successul or NIL if <nPos> is invalid.

Description:

Inserts a column   at <nPos>.

You must call confiure() and refreshAll() to update the browse.

Example:

```
// Add new column for birthday...
oCol := WBColumn{ "Birthday", {||CHILD->DOB} }
oB:insColumn( 2, oCol )
oB:configure()
oB:refreshAll()
```

WBrowse:left()Method

Move browse cursor one column to the left.

Syntax:

left() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Moves the browse cursor left one data column.   If the cursor is already at the right left edge the display is panned if there are additional columns not currently in view.

Example:

```
oB:left() // Move one column to the left...
```

WBrowse:modColumn()Method

Modify an existing browse column.

Syntax:

modColumn(<nCol>, <oCol>) -> **Self**

Arguments:

<nCol> is the column to modify.   <oCol> is a valid WBColumn object with the new column properties.

Returns: Self if <nCol> is valid, NIL if not.

Description:

modColumn() is used to modify a column in an active browse. Once a column has been modified you **MUST** call configure() and refreshAll().

Example:

```
//Modify column based on user input...

oOldCol := oB:getColumn( 2 )
oNewCol := UserModCol( oOldCol )
oB:modColumn( 2, oNewCol )
oB:configure()
oB:refreshAll()
```

WBrowse:panend() Method

Move to last column.

Syntax:

panEnd() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Moves the browse cursor to the rightmost data column causing the display to be panned completely to the right.   Not applicable when autoLite is .T.

Example:

```
oB:panEnd()  // Move to last column...
```

WBrowse:panHome()Method

Move to first column.

Syntax:

panHome() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Moves the browse cursor to the leftmost data column causing the display to be panned completely to the right.   Not applicable when autoLite is .T.

Example:

```
oB:panHome() // Move to first column...
```

WBrowse:panLeft()Method

Pan left one column.

Syntax:

panLeft() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Pans the display to the left one column.

Example:

```
oB:panLeft() // Pan the dispaly one column to the left...
```

WBrowse:panRight()Method
Pan one column to the right.

Syntax:

panRight() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Pans the display to the right one column.

Example:

```
oB:panRight()  // Pan one column to the right...
```

WBrowse:pageDown()Method

Move down one page.

Syntax:

pageDown() - > **Self**

Arguments:

None.

Returns:

Self.

Description:

Repositions the database downward nMaxLines and refills the display.

Example:

```
oB:pageDown() // Move down one page...
```

WBrowse:pageUp() Method

Move up one page.

Syntax:

pageUp() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Repositions the data source upward nMaxLines and refills the display.

Example:

```
oB:pageUp() // Up one page...
```

WBrowse:refreshAll()Method
Refresh and re-display entire browse.

Syntax:

refreshAll() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Forces a complete redraw of the browse window and update of the underlying data.

Example:

```
oB:refreshAll() // Update and redraw...
```

WBrowse:refreshCol()Method

Update column and re-draw it.

Syntax:

refreshCol(<nCol>) -> **Self**

Arguments:

<nCol> is the column to refresh.

Returns:

Self if <nCol> is valid, NIL if not.

Description:

Used to refresh the contents of a single browse column.   Updates the data and display.

Example:

```
oB:refreshCol( 2 ) // Refresh column 2...
```

WBrowse:refreshCurrent() Method

Refresh and re-draw current row.

Syntax:

refreshCurrent() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Forces the current row to be updated.   Should be used with caution since changes in key fields can cause the browse to become unstable in that the data displayed and the record pointers do not match when a key/index value has been changed.   If records key/index field(s) have been changed or a record has been deleted then refreshAll() should be used instead.

Example:

```
oB:refreshCurrent() // Update the current row...
```

WBrowse:right()Method

Move right one column.

Syntax:

right() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Moves the browse cursor right one data column.   If the cursor is already at the right edge the display is panned if there are additional columns not currently in view.

Example:

```
oB:right() // Move right one column...
```

WBrowse:setColBitMap()Method
Set/change bitmap for a column heading.

Syntax:

setColBitMap(<nCol>, <nBitMap>, [<nAlign>]) -> **Self**

Arguments:

<nCol> is the column number.   <nBitMap> is the numeric id of the bitmap.   <nAlign> is the alignment to use - valid values are DT_LEFT, DT_CENTER, or DT_RIGHT.   Alignment defaults to DT_CENTER.

Returns:

Self.

Description:

setColBitMap() is used to specifiy a bitmap to be used in a column heading. Bitmaps **MUST** be included in the application's resource file and assigned a numeric id number.   If the value of <nCol> is 0 then the nubHBitmap and nubHBitAlign are set/changed.

Example:

```
// Set column heading bitmap and alignment...
oB:setColBitmap( 2, ID_SEX, DT_LEFT )
oB:refreshAll()
```

See Also: **modColumn**

WBrowse:setColCargo()Method

Set cargo for a column.

Syntax:

setColCargo(<nCol>, <xCargo>) -> **Self**

Arguments:

<nCol> is the column number.   <xCargo> is any data type to place in the column cargo hold.

Returns:

Self.

Description:

setColCargo() is used to set the cargo for a column.   The column cargo can be any Clipper data type.   The cargo can be used to determine what column a user is in when they have rearragned the columns.

Example:

```
oB:setColCargo( 1, 1 ) // Make sure we can track our columns...
```

See Also: **modColumn**

WBrowse:setColFont()Method

Set the font to be used for a column.

Syntax:

setColFont(<nCol>,   <hFont>) -> **Self**

Arguments:

<nCol> is the column number.   <hFont> is a handle to a valid font.

Returns:

Self if <nCol> is valid, NIL if not.

Description:

setColFont() is used to set the font to be used when drawing a column. <hFont> must be a handle to a valid font.   WBrowse() does not automatically destroy the font so it is up to you to do so when it is no longer needed. You must call configure() and refreshAll() to update the browse.

Example:

```
// Change column font...

hCFont := CreateFont( aCFont )
oB:setColFont( 2, hCFont )
oB:configure()
oB:refreshAll()
```

See Also: **modColumn**

WBrowse:setColPicture()Method

Set/change the picture clause for a column.

Syntax:

setColPicture(<nCol>, <cPicture>) -> **Self**

Arguments:

<nCol> is the column number.   <cPicture> is any valid picture clause used by the Clipper TRANSFORM() function.

Returns:

Self if <nCol> is valid, NIL if not.

Description:

setColPicture() is used to set or change the picture clause for a column. You must call configure() and refreshAll() afterwards.

Example:

```
// Change column picture clause...

oB:setColPicture( 2, "99,999.99" )
oB:configure()
oB:refreshAll()
```

See Also: **modColumn**

WBrowse:setColWidth() Method

Set the width of a column.

Syntax:

setColWidth(<nCol>, <nWidth>) -> **Self**

Arguments:

<nCol> is the column number.   <nWidth> is the new width of the column in pixels.

Returns:

Self if <nCol> is valid, NIL if not.

Description:

Sets the width of column specified by <nCol> to <nWidth>. <nWidth> is the new column width in pixels.   If you use this function you should set autoSize to .F.

By passing 0 for <nCol> you can set/change the value of nubWidth.

You must call configure() and refreshAll() afterwards.

Example:

```
// Change the size of column 2...

oB:setColWidth( 2, 48 )
oB:configure()
oB:refreshAll()
```

See Also: **modColumn**

WBrowse:setHeadBlock()Method
Set/change column heading block.

Syntax:

setHeadBlock(<nCol>, <bBlock>) -> **Self**

Arguments:

<nCol> is the column number.   <bBlock> is the new code block.

Returns:

Self if <nCol> is valid, NIL if not.

Description:

Used to set a codeblock to be evaluated when the user clicks on the column heading.   When this is set for a column the column header will act like a button when clicked on.

If <nCol> is 0 then nubHeadBlock is set to <bBlock>.

Example:

```
// Set heading block for column 2...

oB:setHeadBlock( 2, ||FindKid() )
```

See Also: **modColumn**

WBrowse:setHeadColor()Method

Set/change column heading text color.

Syntax:

setHeadColor(<nCol>, <nColor>) -> **Self**

Arguments:

<nCol> is the column number.   <nColor> is a valid RGB() value to be used when drawing the column heading.

Returns:

Self if <nCol> is valid, NIL if not.

Description:

setHeadColor() is used to set the text color for a column heading.
To set the color for the "nub" heading set <nCol> to 0.

You should call refreshAll() to update the browse.

Example:

```
// Change the column heading color...

oB:setHeadColor( 2, RGB( 128, 0, 0 ) )
oB:refreshAll()
```

See Also: **modColumn**

WBrowse:setHeadHeight()Method

Set height of column headings.

Syntax:

setHeadHeight(<nHeight>) -> **Self**

Arguments:

<nHeight> is the new height in pixels.

Returns:

Self.

Description:

setHeadHeight() is used to manually adjust the height of the browse column headings. In order to use this method you **MUST** set autoSize to **.F.**

You must call configure() and refreshAll() afterwards.

Example:

```
// Change column heading height...

oB:setHeadHeight( 24 )
oB:configure()
oB:refreshAll()
```

See Also: **modColumn**

WBrowse:setRowHeight() Method

Manually set browse row height.

Syntax:

setRowHeight(<nHeight>) -> **Self**

Arguments:

<nHeight> is the new row height in pixels.

Returns:

Self.

Description:

setRowHeight() is used to manually adjust the height of the browse rows. In order to use this method you **MUST** set autoSize to **.F.**

You must call configure() and refreshAll() afterwards.

Example:

```
// Set browse row height based on selected font...

oB:setRowHeight( GetTextHeight( oB:hWnd, hRFont ) )
oB:configure()
oB:refreshAll()
```

WBrowse:showData()Method
Re-displays data currently in browse buffers.

Syntax:

showData() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Re-reads and re-displays the data without re-drawing the column headings. Faster than refreshAll().

Example:

```
// Update browse after an edit...

oB:showData()
```

WBrowse:skip()Method

Moves browse record pointer.

Syntax:

skip(<nRecs>) -> **Self**

Arguments:

<nRecs> number of records to skip. Positive values skip forward/down and negative values skip backwards/up.

Returns:

Number of records skipped.

Description:

Moves browse record pointer up/down <nRecs>.

Evaluates the skipBlock.

Example:

```
oB:skip( 3 ) // Move 3 records down...
```

WBrowse:top()Method

Positions browse at top of file/array.

Syntax:

top() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Re-positions the browse record pointer at the top of the file/array being browsed.   Evaluates goTopBlock.

Example:

```
oB:top()  // Go to top...
```

WBrowse:up()Method

Move browse cursor up one row.

Syntax:

up() -> **Self**

Arguments:

None.

Returns:

Self.

Description:

Moves the browse cursor up one row.   If the cursor is already positioned on the top row then the display is scrolled down.

Example:

```
oB:up()  // Move up one row...
```

WBColumn() Class

Properties:

align

block
bitmap
bitalign
bgcolor

cargo

fgcolor
font

headblock
headcolor
heading

picture

width

Methods:

init

WBColumn:init() Method
Construct a new WBColumn object.

Syntax:

WBColumn{ [<heading>], [<block>], [<picture>], [<fgcolor>], [<bgcolor>], [<align>], [<width>], [<bitmap>], [<bitalign>], [<headblock>], [<cargo>], [<font>], [<headcolor>] } -> **WBColumn object**

Arguments:

[<heading>] - Heading for the column. For multiple heading lines this is an array containing a string for each line.

[<block>] - Code block for extracting the data for the column.

[<picture>] - Optional picture clause. Must conform to the requirements of the CA-Clipper TRANSFORM() function.

[<fgcolor>] - Optional code block that returns a valid RGB() value for the foreground color for the data to be displayed. Defaults to the value specified in colorspec for the un-selected foreground color.

[<bgcolor>] - Optional code block that returns a valid RGB() value for the background color for the data to be displayed. Defaults to the value specified in colorspec for the un-selected background color.

[<align>] - Optional value that specifies how the data will be aligned within the column. Valid values are DT_LEFT, DT_CENTER, and DT_RIGHT. Defaults to DT_LEFT.

[<width>] - Optional column width (in pixels).

[<bitmap>] - Optional bitmap id to be used in the column heading. The bitmap MUST be included in your resource file.

[<bitalign>] - Optional alignment for column heading bitmap. Use DT_LEFT, DT_CENTER, or DT_RIGHT. Defaults to DT_CENTER.

[<headblock>] - Optional codeblock to be evaluated when the user clicks on this column's heading.

[<cargo>] - Optional cargo for the column. This can be any data type. Can also be set using setColCargo() and retrieved using getColCargo().

[<font>] - Optional font for the column. Must be a handle to a valid Windows font.

[<headcolor>] - Optional color value to use when drawing the column heading. Must be a valid value returned from RGB().

NOTES: The code block for each column must return a character type unless the picture parameter is used. Dates should be converted using DTOC() or DTOS(). Memo fields can be used but are not recommended due to thier size and the resulting impact on browse performance. If you want to set the column sizes yourself then you **MUST** set autoSize to .F. or saveColSize to .T.

You can use bitmaps in a column in place of normal data.   When using bitmaps the column block **MUST** return a numeric value equal to the id of the bitmap in your resource file or 0.

All parameters are optional.   When no parameters are passed an empty column object is returned.   You can then assign values to the column objects properties individually.

Description:

Returns a column object for use with a WBrowse() object.

Example:

```
// Create a new column object...

oCol           := WBColumn{}
oCol:heading   := "Sex"
oCol:block     := {||IIF( CHILD->SEX == 'M', "Male", "Female" )}
oCol:picture   := NIL
oCol:fgcolor   := {||IIF( CHILD->SEX == 'M', RGB( 0,0,128), RGB( 0, 128,
0 ) )}
oCol:bgcolor   := {||RGB( 255, 255, 255 )}
oCol:align     := DT_CENTER
oCol:width     := 80
oCol:bitmap    := ID_SEX
oCol:bitalign  := DT_LEFT
oCol:headblock := {|oB|EditRecord( oB ) }
oCol:cargo     := 2
oCol:font      := hSFont
oCol:headcolor := RGB( 128, 0, 0 )
```

WBColumn:align (Access/Assign)

Text alignment for column data.   Must be DT_LEFT, DT_CENTER, or DT_RIGHT. Defaults to DT_LEFT.

Example:

```
oCol:align := DT_CENTER // Center data in column...
```

WBColumn:block (Access/Assign)

Code block for retrieving column data.   Must return a character value unless a picture clause is used or you are using bitmaps in the column.

Example:

```
// Display a check mark if child has been selected...

oCol:block := {||IIF( CHILD->SELECTED, OBM_CHECK, 0 ) }
```

NOTE:

If you want to display more than one line of text/data in a browse row then you simply add a carriage return and linefeed between each line of text/data returned from the column block.

```
// Multi-line browse row sample...

oCol          := WBColumn
oCol:heading := "Name"
oCol:block    := {||MakeName()}

#define CRLF  CHR(13)+CHR(10)

FUNCTION MakeName
      LOCAL cName

      cName := TRIM( CHILD->LASTNAME )  + CRLF + ;
               TRIM( CHILD->FIRSTNAME ) + ;
               IIF( !EMPTY( CHILD->MI ), ', ' + CHILD->MI + '.', '')
RETURN( cName )
```

WBColumn:bitmap (Access/Assign)

The numeric id of the bitmap to use in the column heading.   The bitmap must be contained in the applications resource file or be one of the standard Windows bitmaps.

Example:

```
oCol:bitmap := OBM_CHECK // Windows standard check mark bitmap...
```

WBColumn:bitalign (Access/Assign)

The alignment of the heading bitmap.   Must be DT_LEFT, DT_CENTER, or DT_RIGHT. Defaults to DT_CENTER.

Example:

```
oCol:bitalign := DT_LEFT  // Make room for heading text...
```

WBColumn:bgcolor (Access/Assign)

A code block that returns a valid RGB() value for the foreground text color.

Example:

```
oCol:fgcolor := {||RGB( 0, 128, 0 ) }      // Make data green
oCol:bgcolor := {||RGB( 255, 255, 255 ( } // on white...
```

See Also: fgcolor

WBColumn:cargo (Access/Assign)

Any valid Clipper data type.

Example:

```
oCol:cargo := { 1, "{||CHILD->DOB}" } // Column number and code block...
```

WBColumn:fgcolor (Access/Assign)

A code block that returns a valid RGB() value for the background text color.

Example:

```
oCol:fgcolor := {||RGB( 0, 128, 0 ) }     // Make data green
oCol:bgcolor := {||RGB( 255, 255, 255 ( } // on white...
```

See Also: bgcolor

WBColumn:font (Access/Assign)

A handle to a valid Windows font to be used when displaying column data.

Example:

```
oCol:font := hSFont // Use special font for this column...
```

WBColumn:headblock (Access/Assign)

A code block that will be evaluated when the user clicks on the column heading.

Example:

```
oCol:headblock := {|oB|FindChild( oB ) } // Find a kid...
```

WBColumn:headcolor (Access/Assign)

A valid RGB() value to be used when drawing the column heading text.

Example:

```
oCol:headcolor := RGB( 128, 0, 0 ) // Make heading text red on gray...
```

WBColumn:heading (Access/Assign)

Text to be used for column heading.   For multi-line headings an array is passed.

Example:

```
oCol:heading := { "Date", "Enrolled" } // Column heading...
```

WBColumn:picture (Access/Assign)

A string containing a valid picture for use with the Clipper TRANSFORM() function.

Example:

```
oCol:picture := "99,999.99" // Format numbers this way...
```

WBColumn:width (Access/Assign)

Width in pixels for the column.   You must set autoSize to .F. or saveColSize to .T. to prevent WBrowse() from overriding this value.

Example:

```
// Set column width manually...

oB:saveColSize := .T.
oCol:width     := 80
```

Bitmaps

WBrowse() supports the use of bitmaps in column headings, on the "nubs", and in browse columns.   Bitmaps **MUST** be included in your applications resource file and must use numeric id's.   You can also use the internal Windows bitmaps (OBM_CHECK, etc.).

You can specify bitmaps for column headings in the column object or by calling setColBitMap().

To use bitmaps on the "nubs" the nubBlock **MUST** return the numeric id of the bitmap to use.

Vertical Scroll Bars

When the folks at PARC came up with the vertical scroll bar they created one **BIG** headache for database application developers! It is an almost impossible task to accurately display the thumb in the correct position. Depending on the approach taken you can come close but the presence of deleted records in a database and in it's associated index(s) will defeat most approaches.

The exported instance variable nCurRec is used to keep track of the current record position within WBrowse(). When you use your own skipBlock function you are responsible for setting this variable.

If you use a function that performs a seek operation you will need to update nCurRec once the seek has been performed. You can use NtxPos() or if you are using the SIXCDX RDD you can use Sx_KeyNo().

In an attempt to keep the scrollbar thumb accruately positioned WBrowse() uses a rather dumb approach which is very slow with large databases. If you are using a RDD that provides functions to traverse the index file then you can speed things up a whole lot! The following code shows how to do this using the SIXCDX RDD:

```
#define TRUE  .T.
#define FALSE .F.

STATIC FUNCTION SpeedSkip( oB, nSkip, cAlias )
      LOCAL lOk := TRUE

//     The only time WBrowse() requests a skip in either direction that is
//     greater than the number of visible records is when the user moves
//     the vertical scrollbar thumb

      IF ( nSkip > oB:nMaxLines )
         lOk := SX_KeyGoTo(,,nSkip )
      ELSE
        ( oB:alias )->( DBSKIP( nSkip ) )
      ENDIF
      DO CASE
         CASE ( oB:alias )->( BOF() )
              oB:nCurRec    := 1
              oB:nCrecNo    := ( oB:alias )->( RECNO() )
              oB:hitTop     := TRUE
              oB:hitBottom  := ( oB:alias )->( EOF() )
         CASE ( oB:alias )->( EOF() ) .OR. !lOk
              oB:nCurRec    := ( oB:alias )->( RECNO() )
              oB:nCrecNo    := ( oB:alias )->( RECNO() ) - 1
              oB:hitBottom  := TRUE
              oB:hitTop     := ( oB:alias )->( BOF() )
         OTHERWISE
              oB:nCurRec    += nSkip
              oB:nCrecNo    := ( oB:alias )->( RECNO() )
              oB:hitTop     := FALSE
              oB:hitBottom  := FALSE
              RETURN( nSkip )
      ENDCASE
RETURN( 0 ) // 0 tells WBrowse() we hit EOF() or BOF()
```

Linking

WBROWSET.LIB **MUST** be linked **AFTER** CLIP4WIN.LIB.

Navigation Keys

The following keys are used internally by WBrowse() to move the browse record pointer/column position:

Home	- Moves the record pointer to the first record <**goTop()**>.
End	- Moves the record pointer to the last record <**goBottom()**>.
Alt+Home	- Pans the display to the left most column <**panHome()**>.
Alt+End	- Pans the display to right most column <**panEnd()**>.
PgUp	- Moves the record pointer up one page <**pageUp()**>.
PgDn	- Moves the record pointer down one page <**pageDown()**>.
Up Arrow	- Moves the record pointer up one record <**up()**>.
Down Arrow	- Moves the record pointer down one record <**down()**>.
Left Arrow	- Moves the cursor left one column <**left()**>.
Right Arrow	- Moves the cursor right one column <**right()**>.
Enter	- It evaluates the doubleClick code block if not NIL.
F2	- It evaluates the searchBlock code block if not NIL.

You can use either keyBlock or msgBlock to override these keys.

Icons

WBrowse() will use the default Windows application icon for browse windows that are minimized.   If you want the windows to use a special icon or your applications icon then you must call UnregisterClass() to unregister the WBrowse() window class, then call RegisterClass() to re-register the class with your icon.   The class name used by WBrowse() is "BLIST".   The following code is from the demo program and shows how this all looks:

```
// We want our browses to have our icon so we get rid of the
// BLIST class and re-create it with our apps icon

UnregisterClass("BLIST",hInst)

IF !RegisterClass(CS_HREDRAW+CS_VREDRAW+CS_SAVEBITS+CS_DBLCLKS,;
                 hInst, ;
                 hIcon, ;
                 LoadCursor(,IDC_ARROW),;
                 hBrush,;
                 "BLIST")

  QUIT
ENDIF
```

If you are happy with the default icon then you don't need to do any of the above. WBrowse() contains an INIT procedure which automatically registers the class when your application starts.

You **DO NOT** have to call UnregisterClass() for "BLIST" when your application exits since WBrowse() contains an EXIT procedure which does this for you.

Technical Support

WBrowse() users may obtain technical support via phone, fax or CIS Mail.   Technical support is free of charge except for the cost of a stamp, phone call, or fax transmission!   Phone support is available between 08:00 and 17:00 EST - Monday thru Friday except for national holidays.   Technical support contact numbers:

Logical Systems
2635 Portsmouth Place
Hephzibah, GA 30815   USA

Phone: (706) 790-7303
FAX: (706) 790-7084
CIS: 73257,2066

For those of you who prefer to use Class(y) or HighClass instead of TopClass you can obtain a copy of WBrowse(s) which supports these products from the following:

S & A PC Solutions, Inc.
24 Lena Road
Forestburg, New York 12777

Phone: (914) 794-4647
FAX: (914) 794-6447
CIS: 71650,2251

Please contact them for current pricing and ordering information.