Visual Clipper via GTWVW

**WVWClip Documentation**Copyright 2016-2022 Ashfaq Sial

## Contents

[Contents 2](#_Toc120772249)

[Background 3](#_Toc120772250)

[Build Instructions 6](#_Toc120772251)

[@...BOX 7](#_Toc120772252)

[@...GET 8](#_Toc120772253)

[@...GET CHECKBOX 9](#_Toc120772254)

[@...GET EDITBOX 10](#_Toc120772255)

[@...GET LISTBOX 11](#_Toc120772256)

[@...GET PUSHBUTTON 12](#_Toc120772257)

[@...GET RADIOGROUP 13](#_Toc120772258)

[@...GET TBROWSE 14](#_Toc120772259)

[@...IMAGE 16](#_Toc120772260)

[@...LABEL 17](#_Toc120772261)

[@...LABELOBJ 18](#_Toc120772262)

[@...SAY 19](#_Toc120772263)

[@...SAY PUSHBUTTON 20](#_Toc120772264)

[WMENU TO 21](#_Toc120772265)

[WAlert() 22](#_Toc120772266)

[WClear() 23](#_Toc120772267)

[WClose() 24](#_Toc120772268)

[WInit() 25](#_Toc120772269)

[WOpen() 26](#_Toc120772270)

[WSetup() 27](#_Toc120772271)

## Background

I began with the following Clipper 5.3 code. It functions as expected when compiled using Harbour.

#include "inkey.ch"

PROCEDURE main()

LOCAL cName := 'Fred Bloggs Jr.'

LOCAL dDOB := Date() - 20000

LOCAL cStatus := 'Single'

LOCAL nSalary := 95000

LOCAL lEmployed := .T.

LOCAL lOK := ' '

LOCAL lCancel := ' '

SET CENTURY ON

SetMode( 10, 32 )

CLS

@ 00, 01 SAY 'Employee Details'

@ 02, 12 GET cName CAPTION 'Name:' PICTURE '@K'

@ 03, 12 GET dDOB CAPTION 'Birth Date:' PICTURE '@D'

@ 04, 12, 08, 21 GET cStatus LISTBOX { 'Unknown', 'Married', 'Single' } ;

DROPDOWN CAPTION 'Status:'

@ 05, 12 GET nSalary CAPTION 'Salary:' PICTURE '999,999'

@ 06, 12 GET lEmployed CHECKBOX CAPTION 'Employed?'

@ 08, 12 GET lOK PUSHBUTTON CAPTION 'OK' ;

STATE {|| hb\_keyPut( K\_CTRL\_W ) }

@ 08, 20 GET lCancel PUSHBUTTON CAPTION 'Cancel' ;

STATE {|| hb\_keyPut( K\_ESC ) }

READ

IF LastKey() <> K\_ESC

Alert( cName + ';' + ;

DToC( dDOB ) + ';' + ;

cStatus + ';' + ;

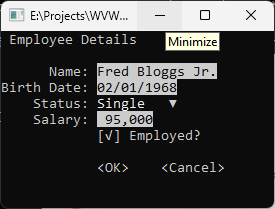
Str( nSalary ) + ';' + ;

iif( lEmployed, 'Y', 'N' ) )

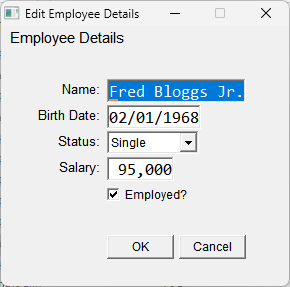
ENDIF

RETURN

The resulting output looks like this.



After making a few changes, highlighted in code shown below , the output looks much better.



#include "inkey.ch"

#include "wvwclip.ch"

PROCEDURE main()

LOCAL cName := 'Fred Bloggs Jr.'

LOCAL dDOB := Date() - 20000

LOCAL cStatus := 'Single'

LOCAL nSalary := 95000

LOCAL lEmployed := .T.

LOCAL lOK := ' '

LOCAL lCancel := ' '

SET CENTURY ON

SET EVENTMASK TO INKEY\_ALL

SetMode( 10, 32 )

WSetup( .T. )

WInit( 'Edit Employee Details' )

@ 00, 01 LABEL 'Employee Details' FONT HeadFont()

@ 02, 12 GET cName CAPTION 'Name:' PICTURE '@K'

@ 03, 12 GET dDOB CAPTION 'Birth Date:' PICTURE '@D'

@ 04, 12, 08, 21 GET cStatus LISTBOX { 'Unknown', 'Married', 'Single' } ;

DROPDOWN CAPTION 'Status:'

@ 05, 12 GET nSalary CAPTION 'Salary:' PICTURE '999,999'

@ 06, 12 GET lEmployed CHECKBOX CAPTION 'Employed?'

@ 08, 12 GET lOK PUSHBUTTON CAPTION 'OK' ;

STATE {|| hb\_keyPut( K\_CTRL\_W ) }

@ 08, 20 GET lCancel PUSHBUTTON CAPTION 'Cancel' ;

STATE {|| hb\_keyPut( K\_ESC ) }

READ

IF LastKey() <> K\_ESC

Alert( cName + ';' + ;

DToC( dDOB ) + ';' + ;

cStatus + ';' + ;

Str( nSalary ) + ';' + ;

iif( lEmployed, 'Y', 'N' ) )

ENDIF

RETURN

## Build Instructions

Add the additional lines of code as highlighted above.

Create wclip.hbp with the following lines in it.

wvwclip.hbc   
hbxpp.hbc   
hbct.hbc   
hbwin.hbc   
gtwvw.hbc

wclip.prg

Compile.

Hbmk2 wclip.hbp

**Note: WVWClip library was built in Harbour 3.4.0dev development environment.**

## @...BOX

Draw a box at given coordinates.

Syntax

**@ <nTop>, <nLeft>, <nBottom>, <nRight> BOX ;**

**RAISED | RECESSED | GROUP ;**

**[CAPTION <cCaption>] ;**

**[OFFSET <aTLBR>]**

Arguments

<cCaption>

Caption text.

<aTLBR>

This is an array of four elements, {nTop, nLeft, nBottom, nRight}, in pixels used for row and column alignment.

Example

@ 0, 0, MaxRow(), MaxCol() BOX RECESSED

@ 4, 23, 7, 33 BOX GROUP ;

CAPTION 'Database'

## @...GET

Draw a data entry field at given coordinates.

Syntax

**@ <nRow>, <nCol> GET <v> ;**

**[CAPTION <cCaption>] ;**

**[PICTURE <cPicture>] ;**

**[VALID <lPostExpression>] ;**

**[WHEN <lPreExpression>] ;**

**[SEND <msg>]**

Example

LOCAL nSalary := 95000

@ 05, 12 GET nSalary CAPTION 'Salary:' PICTURE '999,999' ;

VALID nSalary >= 0

## @...GET CHECKBOX

Draw a check box at given coordinates.

Syntax

**@ <nRow>, <nCol> GET <lVar> CHECKBOX ;**

**[CAPTION <cCaption>];**

**[WHEN <lPreExpression>] ;**

**[VALID <lPostExpression>]**

Arguments

Example

LOCAL lEmployed := .T.

@ 06, 12 GET lEmployed CHECKBOX ;

CAPTION 'Employed?'

## @...GET EDITBOX

Draw an edit box at given coordinates.

Syntax

**@ <nTop>, <nLeft>, <nBottom>, <nRight> GET <cVar> EDITBOX ;**

**[CAPTION <cCaption>] ;**

**[FONT <aFont> ;**

**[STYLE <nStyle>] ;**

**[MAXCHAR <nMaxChar>] ;**

**[OFFSET <aTLBR>]**

Arguments

<nTop>, <nLeft>, <nBottom>, <nRight>

Edit box coordinates.

<cVar>

This is the edited variable initialized as a character string.

<cCaption>

Caption text. It is drawn so that it ends at <nLeft>-1 column position.

<aFont>

Array of two elements: {cFont, nSize}. Specifies which font will be used for this edit box.

<nStyle>

This could be any combination of ES\_\* constants. See example below.

<nMaxChar>

Maximum number of input characters accepted.

<aTLBR>

This is an array of four elements, {nTop, nLeft, nBottom, nRight}, in pixels used for row and column alignment.

Example

LOCAL cMemo := 'These are employee notes.'

@ 0, 5, 6, 35 GET cMemo EDITBOX CAPTION 'Notes:' ;

STYLE ES\_MULTILINE+ES\_READONLY

## @...GET LISTBOX

Draw a dropdown type list box at given coordinates.

Syntax

**@ <nTop>, <nLeft>, <nBottom>, <nRight> GET <nVar|cVar> LISTBOX ;**

**<aOptions> ;**

**[DROPDOWN] ;**

**[CAPTION <cCaption>] ;**

**[OFFSET <aTLBR>]**

Arguments

<nTop>, <nLeft>, <nBottom>, <nRight>

LISTBOX coordinates.

<nVar|cVar>

This is the edited variable initialized as a numeric value or a character string.

If nVar, it is assigned the index (first item is 1) of the selected item of <aOptions>.

If cVar, it is assigned the string of the selected item of < aOptions>.

<aOptions>

An array of options: Each option length must be > 2.

<cCaption>

Caption text. It is drawn so that it ends at <nLeft>-1 column position.

<aTLBR>

This is an array of four elements, {nTop, nLeft, nBottom, nRight}, in pixels used for row and column alignment.

Example

LOCAL cStatus := ‘Single’

@ 04, 12, 04, 19 GET cStatus ;

LISTBOX { 'Unknown', 'Married', 'Single' } ;

CAPTION 'Status:'

## @...GET PUSHBUTTON

Draw a pushbutton at given coordinates.

Syntax

**@ <nRow>, <nCol> GET <lVar> PUSHBUTTON ;  
 [CAPTION <cCaption>] ;**

**[WIDTH <nWidth>] ;**

**[VALID <valid>] ;  
 [WHEN <when>] ;**

**[STATE <bAction>] ;**

**[OFFSET <aTLBR>]**

Arguments

<nRow>, <nCol>

Pushbutton coordinates.

<lVar>

This is a placeholder for the underlying GET. It is always set to True.

<cCaption>

Caption text is used for button label.

<nWidth>

Pushbutton width. It defaults to 7.

<bAction>

This code block is evaluated when pushbutton is clicked by user.

<aTLBR>

This is an array of four elements, {nTop, nLeft, nBottom, nRight}, in pixels used for row and column alignment.

Example

LOCAL lSave := .T.

LOCAL nBtn := 0

@ 08, 02 GET lSave PUSHBUTTON ;

CAPTION 'OK' ;

STATE {|| nBtn := 1, hb\_keyPut( K\_CTRL\_W ) }

## @...GET RADIOGROUP

Draw a group of radio buttons at given coordinates.

Syntax

**@ <nTop>, <nLeft>, <nBottom>, <nRight> GET <nVar> ;**

**RADIOGROUP ;**

**[CAPTION <cCaption>] ;**

**[OFFSET <aTLBR>]**

Arguments

Same as those for Clipper except for the additional ones below.

<cCaption>

Caption text.

<aTLBR>

This is an array of four elements, {nTop, nLeft, nBottom, nRight}, in pixels used for row and column alignment.

Example

LOCAL nRB

LOCAL aRadio[ 3 ]

aRadio[ 1 ] := "Red"   
aRadio[ 2 ] := "Green"   
aRadio[ 3 ] := "Blue"

nRB := 2 // Default radio button

@ 2, 8, 4, 15 GET nRBRADIOGROUP aRadio ;

CAPTION "Colour"

## @...GET TBROWSE

Display browse object on screen.

Syntax

**@ <nTop>, <nLeft>, <nBottom>, <nRight> GET <idVar> ;**

**TBROWSE <oBrowse> ;**

**[WHEN <lPreExpression>] ;**

**[VALID <lPostExpression>] ;**

**[SEND <msg>] ;**

**[GUISEND <guimsg>]**

Arguments

Same as those for Clipper version except that MESSAGE clause is not implemented.

Example

#include "inkey.ch"

#include "wvwclip.ch"

REQUEST DBFCDX

REQUEST HB\_CODEPAGE\_FRWIN

FUNCTION Main()

LOCAL lDummy := .F.

LOCAL oBrowse := NIL

PRIVATE GetList := {}

SET DEFAULT TO .\data

SET EVENTMASK TO INKEY\_ALL

hb\_cdpSelect( "FRWIN" )

hb\_langSelect( "EN" )

SetMode( 7, 30 )

WSetup( .T. )

WInit('Browse Example')

USE currency INDEX currency VIA "DBFCDX" SHARED

// Setup browse.

oBrowse := TBrowse():New()

oBrowse:colorspec := 'N/W\*, W/B\*'

oBrowse:autoLite := .F.

// Add columns.

oBrowse:AddColumn( TBColumn():New( 'ID', { || currency->currid } ) )

oBrowse:AddColumn( TBColumn():New( 'Currency', { || currency->descript } ) )

oBrowse:AddColumn( TBColumn():New( 'Symbol', { || currency->symb } ) )

@ 1, 1, 5, 28 GET lDummy TBROWSE oBrowse GUISEND forceStable()

READ

USE

RETURN NIL

## @...IMAGE

Draw an image at given coordinates.

Syntax

**@ <nTop>, <nLeft>, <nBottom>, <nRight> IMAGE <cFile> ;**

**[OFFSET <lTight>|<aTLBR>] TBITMAP**

Arguments

<nTop>, <nLeft>, <nBottom>, <nRight>

IMAGE coordinates.

<cFile>

Image filename or index into an image list.

<lTight>|<aTLBR>

lTight When set to .T. the image is placed snuggly within the image coordinates.

aTLBR is an array of four elements, {nTop, nLeft, nBottom, nRight}, in pixels used for row and column alignment.

<TBITMAP>  
Image is a transparent bitmap.

Example

@ 1, 1, 3, 7 IMAGE hb\_DirBase() + "logo.bmp" TBITMAP

## @...LABEL

Draw a label at given coordinates.

Syntax

**@ <nRow>, <nCol> LABEL <cLabel> ;**

**[FONT <aFont>] ;**

**[RIGHT] ;**

**[COLOR <cClr>]**

Arguments

<nRow>, <nCol>

Label coordinates.

<cLabel>

Label text.

<aFont>

Array of three elements: {cFont, nSize, nWeight }. Specifies which font will be used to draw the label with.

Default is “Arial”, 16, FW\_NORMAL.

RIGHT

If specified, the expression output will end at <nCol> position.

<cClr>

Output colour in ‘N/W’ format.

Example

LOCAL aFont := { 'Arial', 15, FW\_NORMAL }

@ 02, 05 LABEL 'Use at your own risk.' ;

FONT aFont

## @...LABELOBJ

Draw a label object at given coordinates.

Syntax

**@ <nTop>, <nLeft>[, <nBottom>[, <nRight>]] LABELOBJ <cLabel> ;**

**[WIDTH <nWidth>] [RIGHT] ;**

**[FONT <aFont>] ;**

**[OFFSET <aTLBR>] ;**

**[COLOR <cClr>]**

Arguments

<nTop>, <nLeft>, <nBottom>, <nRight>

Label coordinates.

<cLabel>

Label text.

<nWidth>

Label width.

RIGHT

If specified, the expression output will be right justified.

<aFont>

Array of three elements: {cFont, nSize, nWeight}. Specifies which font will be used to draw the label with.

Default is {“Arial”, 16, FW\_NORMAL}.

<aTLBR>

This is an array of four elements, {nTop, nLeft, nBottom, nRight}, in pixels used for row and column alignment.

<cClr>

Output colour in ‘N/W’ format.

Example

LOCAL aFont := { 'Arial', 17, FW\_BOLD }

@ 02, 05, 03, 21 LABELOBJ 'Employee Details' WIDTH 25 RIGHT FONT aFont

## @...SAY

Output value of an expression at given coordinates.

Syntax

**@ <nRow>, <nCol> ;**

**SAY <exp> [PICTURE <cPicture>] ;**

**[COLOR <cColorString>] ;**

**[CAPTION <cCaption>]**

Arguments

Same as those for Clipper except for the additional ones below.

<cCaption>

Caption text.

Example

@ 01, 10 SAY 'Killer Application V10.0' ;

CAPTION ‘Application:’

Output

Application: Killer Application V10.0

## @...SAY PUSHBUTTON

Draw a pushbutton at given coordinates. This pushbutton will not be part of GET subsystem.

Syntax

**@ <nRow>, <nCol> SAY PUSHBUTTON ;  
 [CAPTION <cCaption>] ;**

**[WIDTH <nWidth>] ;**

**[STATE <bAction>] ;**

**[OFFSET <aTLBR>]**

Arguments

<nRow>, <nCol>

PUSHBUTTON coordinates. Pushbutton is 7 characters wide.

<cCaption>

Caption text is used for button label.

<nWidth>

Button width. It defaults to 7.

<bAction>

This code block is evaluated when pushbutton is clicked by user.

<aTLBR>

This is an array of four elements, {nTop, nLeft, nBottom, nRight}, in pixels used for row and column alignment.

Example

LOCAL lSave := .T.

LOCAL nBtn := 0

@ 08, 02 SAY PUSHBUTTON ;

CAPTION 'OK' ;

STATE {|| hb\_keyPut( K\_CTRL\_W ) }

## WMENU TO

Returns last menu event of active menu. See WVWMENU.PRG for more details.

Syntax

**WMENU TO <nVar>**

Arguments

<nVar>

The variable receives the last menu event of active menu.

Example

WMENU TO nOpt

## WAlert()

Display a dialog box. This function is not the same as Alert().

Syntax

**WAlert( <cMessage> ), <cHead>] )**

Arguments

<cMessage>

Text to be displayed in the dialog box.

<cHead>

Dialog box heading. Defaults to ‘Alert’.

Example

WAlert( cName + ';' + ;

DToC( dDOB ) + ';' + ;

cStatus + ';' + ;

Str( nSalary ) + ';' + ;

iif( lEmployed, 'Y', 'N' ), ;

‘Warning! ’ )

## WClear()

Clear topmost window and its GUI objects.

Syntax

**WClear()**

Arguments

None

Example

WClear()

## WClose()

Close topmost window.

Syntax

**WClose()**

Arguments

None

Example

WClose()

## WInit()

Initialize current window.

Syntax

**WInit( [<cTitle>] )**

Arguments

<cTitle>

Window title.

Example

WInit( ‘Edit Employee Details’ )

## WOpen()

Open a new sub-window.

Syntax

**WOpen( <nTop>, <nLeft, <nBottom>, <nRight>, ;**

**<cTitle>, <lGUIWin>, <lCenter>, <nLineSpacing> ) ;**

**=> nWinNum**

Arguments

<nTop>, <nLeft, <nBottom>, <nRight>

Sub-window coordinates

<cTitle>

Window title.

<lGUIWin>

.T. – GUI window (GETs will have boxes drawn around them)

.F. – Console window

Defaults to application type specified in WSetup() call.

<lCenter>

Flag to center window. Defaults to .F..

<nLineSpacing>

See GTWVW documentation. It must be an even number defaulting to:

6 for GUI window or

0 for console window.

Example

LOCAL nWinNum

nWinNum := WOpen( 2, 3, 11, 35, ‘Browse Example’, .T., .T., 0 )

## WSetup()

Setup a GTWVW environment. This function is called only once at the start of an application.

Syntax

**WSetup( <lGUIApp>, < cIconFile > )**

Arguments

<lGUIApp>

Set it to .T. for GUI applications. Default .F.; Console application.

<cIconFile>

Application icon file.

Example

WSetup( .T., **hb\_DirBase**()+"resource\wvwclip.ico" )