

PROGRAMMING TURBOCAD USING THE SDK (VBS) **TESTED ON TURBOCAD PROFESSIONAL 11, 12, AND 14**

TurboCAD, like DeltaCAD does offer a programming methodology. However, rather than use one simple system provided by another software vendor as does DeltaCAD, TurboCAD elected to use the set of languages available from Microsoft. Further, the interface is only available with the Professional version, not with the more affordable Deluxe installation. Thus, DeltaCAD is likely to remain the CAD system of choice for the average diallist.

The TurboCAD VBS (Visual Basic Script) macros are to be stored in:-
c:\program files
 IMS\TCWP11 (or whatever release...)
 SDK\Samples\VBS\WshScriptPack
 (Wsh means: GUI Windows Script Host)

In TurboCAD 11 and 12 Professional, folders are found for:-

C#, CPP, DELPHI, VB NET, VBASIC, VBS, vc, and finally vcnet

Several problems come to light at this juncture.

1. MicroSoft languages come and go, and the languages themselves are not stable as they are enhanced, and some features dropped.
2. Most have either involved build processes or collections of libraries that must go with the program, VBS (visual basic script) is the exception.
3. While TurboCAD has an excellent book for the hands on CAD user, the two or three pages referring to their SDK are in no way enlightening.
4. The computer readable documentation is at best descriptive and of use to someone already knowing the system. There is no obvious cross reference between simple functions and their calls, functions, or methods. Thus finding how to draw a line is not intuitively obvious.

While VBS is the slowest interface, it does not require a host of libraries, nor a complex build. Instead, a simple text editor (WordPad for example) can build the program and save it in the appropriate folder, and in TurboCAD Professional, clicking the MACRO tool allows the user to then click on FILE FOLDERS, and from thence the VBS program may be run.

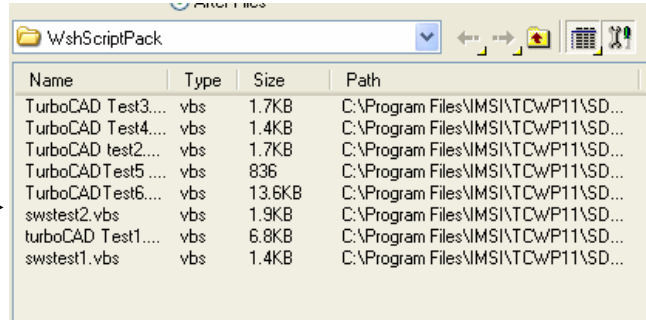
Macros (programs) are executed by loading TurboCAD Professional, clicking on the macro tool (if visible), or VIEW, MacroRecorderPalette (if not),



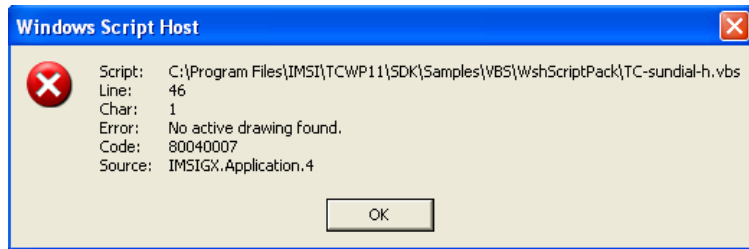
which brings up a sub panel, click on files (and set to VBS if needed)



and then



click on the file you wish to run. NOTE: the drawing area must exist, so do a NEW when bringing up TurboCAD, otherwise the following Windows Script Host message appears.



Useful references sources are:-

<http://msdn2.microsoft.com/en-us/library/ms950396.aspx>
<http://www.tyharness.co.uk/tcvbacrashcourse/tcvbacrashcourse.htm>
<http://www.tyharness.co.uk/cad.htm>

and the file: tcsdk.chm in the Docs folder of the SDK, is not without merit. Cryptic at times, it can provide some clues. And the SEARCH function, once a function or method has been identified, is helpful.

VBS IN TURBOCAD – KEY POINTS TO KNOW

This implementation is object oriented. Thus a graphics area is acquired, and things added to that area, or, object within that area.

AddLineRectangle	adds a rectangle
AddCircleCenterAndPoint	adds a circle
AddLineSingle	adds a line
AddText	adds text into the drawing

graphic elements built, such as a line, have methods that can be invoked

```
Properties("Penstyle").Value = "DASHED"  
Properties("Pencolor").Value = RGB(0, 255, 0)
```

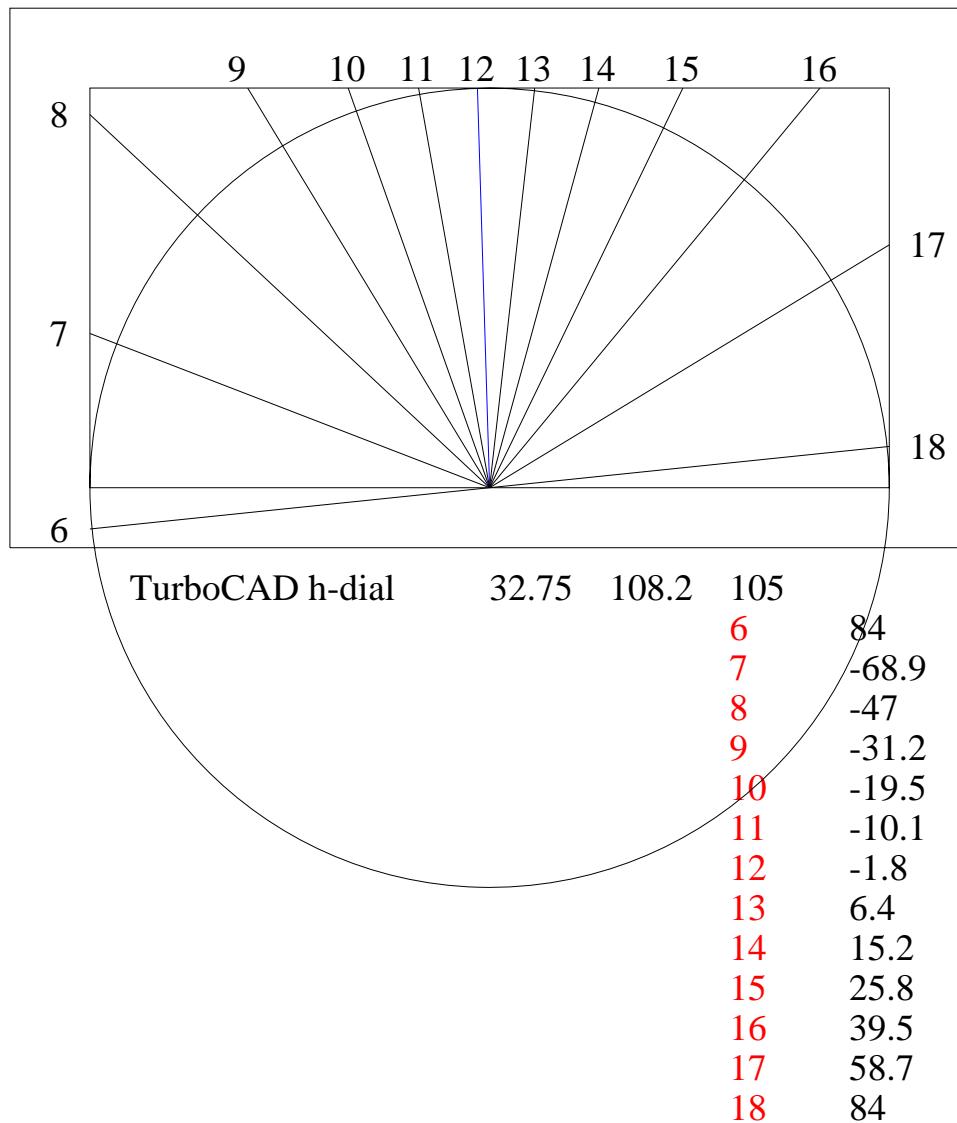
and human interaction comes from:

InputBox	get a line of input
----------	---------------------

Other functions exist including trigonometric functions, however ASN has to be programmed by the programmer, as does the "to" and "from" degrees/radians.

The pictorial below is the result of running the subsequent macro (program) for an h-dial
 There is also a vertical declining dial for TurboCAD which is somewhat different from the
 DeltaCAD version.

H DIAL



```
' *****
' *** Variables etc needed by "the system" do do its thing - June 2007 SWS ***
' *****

Option Explicit

' NOTE: useful references are
' http://msdn2.microsoft.com/en-us/library/ms950396.aspx
' http://www.tyharness.co.uk/tcvbacrashcourse/tcvbacrashcourse.htm
' http://www.tyharness.co.uk/cad.htm
'
' NOTE: The TurboCAD VBS (Visual Basic Script) macros are to be stored in:-
' c:\program files
'       IMSI\TCWP11
'       SDK\Samples\VBS\WshScriptPack      (Wsh means: GUI Windows Script
Host)
'
' NOTE: The TurboCAD help file for the SDK and its functions is located in:-
'       SDK\Docs\tcsdk.chm                (double click this file)
'       CONTENTS                          (click this header)
'       TurboCAD Object Reference          (click this)
'                                           (difficult to find functions)
'       SEARCH                            (enter known function and do search)
' In essence, this help system assumes you already know the functions (methods)
' you desire, it is in no way a tutorial on how to build a VBScript using TurboCAD

'
' However, finding things takes some effort. In essence it is a help file written
' assuming you know the system, it is not a how to at all.
'
' NOTE: VBS (visual basic script) is the slowest but simplest TurboCAD programming system
'
' NOTE: Macros are executed by loading TurboCAD Professional, clicking on the macro tool
'       which brings up a sub panel, click on files
'       and then click on the file you wish to run
' NOTE: the drawing area must exist, so do a NEW when bringing up TurboCAD

' make tcApp an object so that this program can mess with stuff in TurboCAD
Dim tcApp
Set tcApp = CreateObject("TurboCAD.Application")

' get the active Drawing object, i.e. a current open TurboCAD drawing.
Dim ActiveDrawing
Set ActiveDrawing = tcApp.ActiveDrawing
' see: c:\Program Files\IMSI\TCWxxxx\SDK\Docs\tcsdk.chm
' then: Exploring the SDK in Depth
' then: The Application Object
' then: What's in an Application

' invoke the main controlling sub routine below
Sundial

' *****
' *** MAIN CONTROLLING SUB ROUTINE ***
' *****

Sub Sundial()

' dial location parameters
dim lat
dim lng
dim ref

' dial plate issues
dim dvh
dim hr
dim hrlnangle

Dim aline

Dim gxGrs
Dim gxGr

Dim x
Dim y

x = 0
```

```

y = 0

dim toRad
dim toDeg
toDeg = 360/(2 * 3.1416)
toRad = 2*3.1416/360

lat = InputBox("Latitude", "Enter location latitude", 32.75, 100,200)
lng = InputBox("Longitude", "Enter location longitude", 108.2, 100,200)
ref = InputBox("Legal meridian", "Enter standard time legal longitude", 105, 100,200)
dvh = InputBox("Hour divisions", "Enter hour line precision 1,2,4", 1, 10,10)

' use 100 as the basic left/right/top/bottom of the boundary areas
Set gxGrs = ActiveDrawing.Graphics

gxGr = gxGrs.AddText("TurboCAD h-dial", -90, -20, 0, 10)
gxGr = gxGrs.AddText(lat, 0, -20, 0, 10)
gxGr = gxGrs.AddText(lng, 30, -20, 0, 10)
gxGr = gxGrs.AddText(ref, 60, -20, 0, 10)

hr = -6
Do While hr < 6.25

' Rad = (n * 2 * 3.1416) / 360
' Deg = (360 * n) / (2 * 3.1416)

Set gxGr = gxGrs.AddText(12+hr, 60, -30-y, 0, 10) ' NOTE: this Set is
required
hrlnangle = atn(tan(((hr*15)+(ref-lng))*toRad)*sin(lat*toRad))*toDeg
gxGr.Properties("Pencolor").Value = RGB(255,0, 0) ' RED
Set gxGr = gxGrs.AddText(sgn(hrlnangle)*abs(int(hrlnangle*10))/10, 90, -30-y,
0, 10) ' NOTE: this Set is required
y = y + 10

drawhourline hr, hrlnangle
gxGr.Draw

hr = hr + (1/dvh)
Loop

'
*****
' *** The logic for this horizontal dial is found in the DeltaCAD macro for the h-dial
' ***
'
*****
Dim tcgrs
Dim tcGr1
dim tcDwg
Set tcDwg = tcApp.ActiveDrawing
Set tcGrs = tcDwg.Graphics

' create the circle graphic
'
Set tcGr1 = tcGrs.AddCircleCenterAndPoint(x, y, z, x, y, z)
(0, 0, 0, 100, 0, 0)

'
' Set tcGr1 = tcGrs.AddLineSingle x, y, x, y
(0, 0, 0, 80, 20, 0)
' tcGr1.Properties("Penstyle").Value = "DASHED"
' tcGr1.Properties("Pencolor").Value = RGB(0, 0, 255) ' BLUE

'
Set tcGr1 = tcGrs.AddLineRectangle x, y, x, y
(-100, 0, 0, 100, 100, 0)
Set tcGr1 = tcGrs.AddLineRectangle (-120, -15, 0, 120, 120, 0)

ActiveDrawing.ActiveView.ZoomToExtents ' also forces colors etc to display
correctly

MsgBox "Done"

End Sub

' *****
' *** draw an hour line but use constraints ***
' *****

```

```
sub drawhourline (hour,angle)

    dim toRad
    toRad = 2*3.1416/360

    ' use 100 as the basic left/right/top/bottom of the boundary areas
    dim xx, yy

    Dim tcgrs
    Dim tcGr1
    dim tcdwg
    Set tcDwg = tcApp.ActiveDrawing
    Set tcGrs = tcDwg.Graphics

    ' correct angles such as 0600 for west of legal meridian locations
    if hour < 0 then
        if angle > 0 then
            angle = angle - 180
        end if
    end if

    ' correct angles such as 1800 for east of legal meridian locations
    if hour > 0 then
        if angle < 0 then
            angle = 180 - angle
        end if
    end if

    ' hours close to noon
    if abs(angle) <= 45 then
        yy = 100
        xx = yy*tan(toRad*angle)

        Set tcGr1 = tcGrs.AddLineSingle (0,0, 0, xx, yy, 0)

        ' noon is blue
        if hour = 0 then
            tcGr1.Properties("Penstyle").Value = "SOLID"
            tcGr1.Properties("Pencolor").Value = RGB(0, 0, 255) ' BLUE
        end if

        ' non full hours are green
        if hour - int(hour) <> 0 then
            tcGr1.Properties("Penstyle").Value = "DASHED"
            tcGr1.Properties("Pencolor").Value = RGB(0, 255, 0) ' GREEN
        end if

        ' say what hour is associated with this hour line
        if hour - int(hour) = 0 then
            Set tcGr1 = tcgrs.AddText(12+hour, xx-5,yy+10, 0, 10) ' NOTE: this Set
is required
        end if
    end if

    ' hours close to sunrise and sunset
    if abs(angle) > 45 then
        if angle < 0 then
            xx = -100
        else
            xx = 100
        end if

        yy = xx*tan(toRad*(90-angle))

        Set tcGr1 = tcGrs.AddLineSingle (0,0, 0, xx, yy, 0)

        ' noon is blue
        if hour = 0 then
            tcGr1.Properties("Penstyle").Value = "SOLID"
            tcGr1.Properties("Pencolor").Value = RGB(0, 0, 255) ' BLUE
        end if

        ' non full hours are green
        if hour - int(hour) <> 0 then
            tcGr1.Properties("Penstyle").Value = "DASHED"
            tcGr1.Properties("Pencolor").Value = RGB(0, 255, 0) ' GREEN
        end if
    end if
end sub
```

```
' say what hour is associated with this hour line
if hour - int(hour) = 0 then
  if hour < 0 then
    Set tcGr1 = tcgrs.AddText(12+hour,    xx-10,yy+5, 0, 10)      ' NOTE: this
Set is required
  else
    Set tcGr1 = tcgrs.AddText(12+hour,    xx+5, yy+5, 0, 10)      ' NOTE: this
Set is required
  end if
end if
end if

end sub

' *****
' * END *
' *****
```