Stefan's Advanced Polygon API Demos

-
StPendl

Stefan's Advanced Polygon API Demos | Array of Structures | String Hack

Array of Structures

Currently there are two ways to handle an array of structures.

1) using direct memory access. This is the regular way, but involves using API functions.

```
    'draw_polygon.bas
    'Author: Stefan Pendl
    'Date: 30.01.06
    '
    ' draw polygon thru user defined points
    ' minimum number of points is 3
    '
    ' left mouse button to set ponts
    ' right mouse button to daw poly

    nomainwin

    polyDrawn = 0
    ptSize = 4
    ptColor$ = "red"
    polySize = 1
    polyColor$ = "black"
    fillColor$ = "green"
    polyFillColor$ = "blue"

    gbWidth = 300
    gbHeight = 300

    if gbWidth > DisplayWidth - 30 then gbWidth = DisplayWidth - 30
    if gbHeight > DisplayHeight - 90 then gbHeight =
DisplayHeight - 90

    WindowWidth = gbWidth + 30
    WindowHeight = gbHeight + 90

    UpperLeftX = int((DisplayWidth-WindowWidth)/2)
    UpperLeftY = int((DisplayHeight-WindowHeight)/2)

    graphicbox #1.gb, 10, 10, gbWidth, gbHeight
    button #1.draw, "Draw Poly", [drawPoly], ul, 10,
 gbHeight + 20, 60, 20
    stylebits #1.ptnum, _SS_RIGHT, 0, 0, 0
    statictext #1.ptnum, "0", 80, gbHeight + 20, 15, 15
    statictext #1.pt, "Points selected", 100, gbHeight + 20, 100, 15

    statictext #1.ptx, "X:", 200, gbHeight + 20, 10, 15
    stylebits #1.ptxpos, _SS_RIGHT, 0, 0, 0
    statictext #1.ptxpos, "000", 215, gbHeight + 20, 20, 15

    statictext #1.pty, "Y:", 200, gbHeight + 40, 10, 15
    stylebits #1.ptypos, _SS_RIGHT, 0, 0, 0
    statictext #1.ptypos, "000", 215, gbHeight + 40, 20, 15
    open "Polygon" for window_nf as #1
    #1 "trapclose [quit]"
    #1.draw "!disable"
    #1.gb "down;fill ";fillColor$;";flush"
    #1.gb "size "; ptSize
    #1.gb "color "; ptColor$
    #1.gb "backcolor "; polyFillColor$
    #1.gb "when mouseMove [updatePos]"
    #1.gb "when leftButtonUp [drawPoint]"
    #1.gb "when rightButtonUp [drawPoly]"
    h=hwnd(#1.gb) 'window handle

    'get device context for window:
    calldll #user32, "GetDC",_
        h   as ulong,_
        hdc as ulong

    cursor crosshair
    wait

[updatePos]
    #1.ptxpos MouseX
    #1.ptypos MouseY
    wait

[drawPoint]
    if polyDrawn = 1 then
        #1.gb "cls;fill "; fillColor$; ";flush blank"
        polyDrawn = 0
        Points$ = ""
        ptNum = 0
    end if

    Points$ = Points$; MouseX; " "; MouseY; " "

    #1.gb "set "; MouseX; " "; MouseY

    ptNum = ptNum + 1
    #1.ptnum ptNum

    if ptNum = 3 then #1.draw "!enable"
    wait

[drawPoly]
    if ptNum < 3 then wait

    #1.draw "!disable"
    #1.gb "size "; polySize
    #1.gb "color "; polyColor$

    STRUCT PolyPoints,_
        x as long,_
        y as long

    PolyPointsLen = len(PolyPoints.struct)

    cbBuf = PolyPointsLen * ptNum

    uFlags = _LMEM_MOVEABLE or _LMEM_ZEROINIT

    CallDll #kernel32, "LocalAlloc", _
        uFlags  as ulong, _
        cbBuf   as ulong, _
        hMem    as ulong

    CallDll #kernel32, "LocalLock", _
        hMem            as ulong, _
        PolyPointsArray as ulong

    BufferPointer = PolyPointsArray

    'The STRUCT must be filled before it can be used in an api call:
    for count = 1 to ptNum * 2 step 2
        PolyPoints.x.struct = val(word$(Points$, count))
        PolyPoints.y.struct = val(word$(Points$, count + 1))
        calldll #kernel32, "RtlMoveMemory", _
            BufferPointer as ulong, _
            PolyPoints    as struct,_
            PolyPointsLen as ulong, _
            result        as void

        BufferPointer = BufferPointer + PolyPointsLen
    next

    calldll #gdi32, "Polygon",_
        hdc            as ulong,_
'device context of window or control
        PolyPointsArray as ulong,_
'memory address of array of points
        ptNum          as ulong,_     'number of x,y pairs in array
        result         as long

    #1.gb "getbmp pix 0 0 ";gbWidth; " "; gbHeight
    #1.gb "delsegment Polygon;drawbmp pix 0 0;flush Polygon"
    #1.gb "size "; ptSize
    #1.gb "color "; ptColor$
    polyDrawn = 1

    ' free PolyPointsArray memory
    calldll #kernel32, "LocalFree", _
        hMem   as ulong, _
        result as ulong
    wait

[quit]
    calldll #user32, "ReleaseDC",_
        h   as ulong,_
        hdc as ulong,_
        ret as long

    close #1
    end
```

String Hack

This one is based on an easter egg of LB, where a structure is basically handled like a string, since you use the LEN() function to get the size of the structure in LB.

```
    'draw_poly.bas
    'Author: Stefan Pendl
    'Date: 08.07.08
    '
    ' draw polygon thru user defined points
    ' minimum number of points is 3
    '
    ' left mouse button to set ponts
    ' right mouse button to daw poly

    nomainwin

    polyDrawn = 0
    ptSize = 4
    ptColor$ = "red"
    polySize = 1
    polyColor$ = "black"
    fillColor$ = "green"
    polyFillColor$ = "blue"

    gbWidth = 300
    gbHeight = 300

    if gbWidth > DisplayWidth - 30 then gbWidth = DisplayWidth - 30
    if gbHeight > DisplayHeight - 90 then gbHeight =
DisplayHeight - 90

    WindowWidth = gbWidth + 30
    WindowHeight = gbHeight + 90

    UpperLeftX = int((DisplayWidth-WindowWidth)/2)
    UpperLeftY = int((DisplayHeight-WindowHeight)/2)

    graphicbox #1.gb, 10, 10, gbWidth, gbHeight
    button #1.draw, "Draw Poly", [drawPoly], ul, 10,
 gbHeight + 20, 60, 20
    stylebits #1.ptnum, _SS_RIGHT, 0, 0, 0
    statictext #1.ptnum, "0", 80, gbHeight + 20, 15, 15
    statictext #1.pt, "Points selected", 100, gbHeight + 20, 100, 15

    statictext #1.ptx, "X:", 200, gbHeight + 20, 10, 15
    stylebits #1.ptxpos, _SS_RIGHT, 0, 0, 0
    statictext #1.ptxpos, "000", 215, gbHeight + 20, 20, 15

    statictext #1.pty, "Y:", 200, gbHeight + 40, 10, 15
    stylebits #1.ptypos, _SS_RIGHT, 0, 0, 0
    statictext #1.ptypos, "000", 215, gbHeight + 40, 20, 15
    open "Polygon" for window_nf as #1
    #1 "trapclose [quit]"
    #1.draw "!disable"
    #1.gb "down;fill ";fillColor$;";flush"
    #1.gb "size "; ptSize
    #1.gb "color "; ptColor$
    #1.gb "backcolor "; polyFillColor$
    #1.gb "when mouseMove [updatePos]"
    #1.gb "when leftButtonUp [drawPoint]"
    #1.gb "when rightButtonUp [drawPoly]"
    h=hwnd(#1.gb) 'window handle

    'get device context for window:
    calldll #user32, "GetDC",_
        h   as ulong,_
        hdc as ulong

    cursor crosshair
    wait

[updatePos]
    #1.ptxpos MouseX
    #1.ptypos MouseY
    wait

[drawPoint]
    if polyDrawn = 1 then
        #1.gb "cls;fill "; fillColor$; ";flush blank"
        polyDrawn = 0
        Points$ = ""
        ptNum = 0
    end if

    Points$ = Points$; MouseX; " "; MouseY; " "

    #1.gb "set "; MouseX; " "; MouseY

    ptNum = ptNum + 1
    #1.ptnum ptNum

    if ptNum = 3 then #1.draw "!enable"
    wait

[drawPoly]
    if ptNum < 3 then wait
    #1.draw "!disable"
    #1.gb "size "; polySize
    #1.gb "color "; polyColor$

    STRUCT PolyPoints,_
        x as long,_
        y as long

    PolyPointsArray$ = ""

    'The STRUCT must be filled before it can be used in an api call:
    for count = 1 to ptNum * 2 step 2
        PolyPoints.x.struct = val(word$(Points$, count))
        PolyPoints.y.struct = val(word$(Points$, count + 1))

        PolyPointsArray$ = PolyPointsArray$; PolyPoints.struct
    next

    calldll #gdi32, "Polygon",_
        hdc              as ulong,_
'device context of window or control
        PolyPointsArray$ as ptr,_
'memory address of array of points
        ptNum            as ulong,_     'number of x,y pairs in array
        result           as long

    #1.gb "getbmp pix 0 0 ";gbWidth; " "; gbHeight
    #1.gb "delsegment Polygon;drawbmp pix 0 0;flush Polygon"
    #1.gb "size "; ptSize
    #1.gb "color "; ptColor$
    polyDrawn = 1

    wait

[quit]
    calldll #user32, "ReleaseDC",_
        h   as ulong,_
        hdc as ulong,_
        ret as long

    close #1
    end
```

Stefan's Advanced Polygon API Demos | Array of Structures | String Hack