

```

/*****
/*
/*----- M A S T E R 3 2 . C -----*/
/* Task      : Starts slave processes that communicate with the
/*             master through messages.
/*-----*/
/* Author      : Michael Tischer and Bruno Jennrich
/* developed on : 08/25/1995
/* last update  : 09/03/1995
/*****
#include <windows.h>
#include <math.h>

#include "privat.h"

#include "resource.h"

/*= Global variables =====*/
long      lAngle;
COLORREF  crColor;

/*= Constants =====*/
#define CHILD_WIDTH  300
#define CHILD_HEIGHT 300

/*****
/* DialogProc : Window function of the dialog box
/*-----*/
/* Parameter:  default dialog parameters
/* Return value: default dialog return value
/*****
BOOL CALLBACK DialogProc( HWND hWnd, UINT uMsg, WPARAM wP, LPARAM lP )
{
    static int dwX = 0;
    static int dwY = 0;

    switch( uMsg )
    {
        case PM_PROCESSEXIT:          /* Slave process reports termination */
        {
            LONG i, lCnt;
            HWND hListBox = GetDlgItem( hWnd, IDC_PROCESSWINDOWS );
            /* Browse through all list entries to find entry of
            /* slave process and then remove it.
            lCnt = SendMessage( hListBox, LB_GETCOUNT, 0, 0L );

            for( i = 0; i < lCnt; i++ )
                if( SendMessage( hListBox, LB_GETITEMDATA, i, 0L ) == lP )
                    SendMessage( hListBox, LB_DELETESTRING, i, 0L );
        }
        break;

        case PM_PROCESSCREATED:        /* Slave process reports creation */
        {
            char Buffer[ 20 ];
            long lIdx;
            HWND hListBox = GetDlgItem( hWnd, IDC_PROCESSWINDOWS );
            HWND hWndProcess = ( HWND ) lP;

            /* Handle of slave process window in clear text in Listbox ----*/
            wsprintf( Buffer, "0x%lX", lP );
            lIdx = SendMessage( hListBox, LB_ADDSTRING, 0, ( LONG ) Buffer );
            /* Also handle as numeric value in ItemData of the entry
            SendMessage( hListBox, LB_SETITEMDATA, lIdx, lP );

            /* Send Pythagoras parameter to slave process -----*/
            SendMessage( hWndProcess, PM_ANGLE, 0, lAngle );
            SendMessage( hWndProcess, PM_COLOR, 0, crColor );

            /* Cause creation of PainterThread -----*/
            SendMessage( hWndProcess, PM_CREATETHREAD, 0, crColor );
        }
        break;

        case WM_INITDIALOG:            /* Prepare DialogBox */

```

```

{ char Buffer[ 20 ];

    lAngle = 30;                                     /* set defaults */
    crColor = RGB( 0, 0, 0 );

    /* Enter angle in input dialog -----*/
    wsprintf( Buffer, "%ld", lAngle );
    SetWindowText( GetDlgItem( hWnd, IDC_ANGLE ), Buffer );

    return TRUE;
}
break;

case WM_PAINT:                                     /* On repaint, display the selected color */
{
    /* in a rectangle under the color button */
    RECT r;
    POINT p;
    HBRUSH hBR;
    HDC hDC;
    PAINTSTRUCT ps;

    hDC = BeginPaint( hWnd, &ps );

    /* Get rectangle of invisible frame... -----*/
    GetWindowRect( GetDlgItem( hWnd, IDC_COLORFRAME ), &r );

    /* ...and convert screen coordinates to client coordinates -----*/
    p.x = r.left; p.y = r.top;
    ScreenToClient( hWnd, &p );
    r.left = p.x; r.top = p.y;

    p.x = r.right; p.y = r.bottom;
    ScreenToClient( hWnd, &p );
    r.right = p.x; r.bottom = p.y;

    /* Create brush in selected color -----*/
    hBR = CreateSolidBrush( crColor );

    FillRect( hDC, &r, hBR );                       /* Fill rectangle */

    DeleteObject( hBR );                             /* Delete brush */

    EndPaint( hWnd, &ps );
}
break;

case WM_COMMAND:
    switch( LOWORD( wParam ) )
    {
        case IDC_TERMINATE:                         /* Terminate slave process */
        {
            LONG lIdx;
            HWND hListBox = GetDlgItem( hWnd, IDC_PROCESSWINDOWS );
            lIdx = SendMessage( hListBox, LB_GETCURSEL, 0, 0L );
            if( lIdx >= 0 )
            {
                HWND hWndProcess = ( HWND )SendMessage( hListBox,
                                                            LB_GETITEMDATA, lIdx, 0L );
                SendMessage( hWndProcess, PM_EXITPROCESS, 0, 0L );
            }
        }
    }
    break;

case IDOK:                                         /* OK = New process */
{
    char Buffer[ 20 ];
    STARTUPINFO    StartInfo;
    PROCESS_INFORMATION ProcInfo;

    /* Get text from input dialog -----*/
    GetWindowText( GetDlgItem( hWnd, IDC_ANGLE ),
                    Buffer,
                    sizeof( Buffer ) );

    /* Ascii to Long -----*/
    lAngle = atol( Buffer );

```

```

if( ( lAngle < 0 ) || ( lAngle > 90 ) )
{
    MessageBox( hWnd,
        "Angle must amount to a value between 0 and 90 degrees",
        "Error", MB_OK );
    break;
}

/* Write window handle of dialog box to environment variable*/
wsprintf(Buffer, "%ld", hWnd );
SetEnvironmentVariable( "Window", Buffer );

StartInfo.cb = sizeof( STARTUPINFO );
StartInfo.lpReserved = NULL;
StartInfo.lpDesktop = NULL;
StartInfo.lpTitle = NULL;
StartInfo.dwX = dwX;
StartInfo.dwY = dwY;
StartInfo.dwXSize = CHILD_WIDTH;
StartInfo.dwYSize = CHILD_HEIGHT;
StartInfo.dwXCountChars = 0;
StartInfo.dwYCountChars = 0;
StartInfo.dwFillAttribute = 0;
StartInfo.dwFlags = STARTF_USESHOWWINDOW |
                    STARTF_USEPOSITION |
                    STARTF_USESIZE;

StartInfo.wShowWindow = TRUE;
StartInfo.cbReserved2 = 0;
StartInfo.lpReserved2 = NULL;
StartInfo.hStdInput = NULL;
StartInfo.hStdOutput = NULL;
StartInfo.hStdError = NULL;

dwX += 20; /* set next window position */
dwY += 20;

if((dwX+CHILD_WIDTH >= GetSystemMetrics(SM_CXFULLSCREEN)) ||
    (dwY+CHILD_HEIGHT >= GetSystemMetrics(SM_CYFULLSCREEN)))
    dwX = dwY = 0;

/* Start slave process (EXE file in current dir.) -----*/
if( !CreateProcess( "SLAVE32.EXE",
    "",
    NULL,
    NULL,
    FALSE,
    0L,
    NULL,
    NULL,
    &StartInfo,
    &ProcInfo ) )
    MessageBox( hWnd, "Could not create process!",
        "Error", 0 );
}
break;

case IDCANCEL:
    EndDialog( hWnd, 0 ); /* Good-Bye! */
break;

case IDC_COLOR: /* select color */
{
    CHOOSECOLOR cc;
    COLORREF crCustom[ 16 ];
    cc.lStructSize = sizeof( CHOOSECOLOR );
    cc.hwndOwner = hWnd;
    cc.hInstance = NULL;
    cc.rgbResult = crColor;
    cc.lpCustColors = crCustom;
    cc.Flags = CC_PREVENTFULLOPEN | CC_RGBINIT;
    cc.lCustData = 0L;
    cc.lpfnHook = NULL;
    cc.lpTemplateName = NULL;

    if( ChooseColor( &cc ) )
    {
        /* Color selected */

```

```

        crColor = cc.rgbResult;          /* note color... */
        InvalidateRect( hWnd, NULL, FALSE ); /* ..and display */
    }
}
break;
}
break;

case WM_DESTROY:    /* Master terminating, so all slaves will too */
{
    LONG i, lCnt;
    HWND hListBox = GetDlgItem( hWnd, IDC_PROCESSWINDOWS );

    lCnt = SendMessage( hListBox, LB_GETCOUNT, 0, 0L );
    for( i = 0; i < lCnt; i++ )
    {
        HWND hWndProcess = ( HWND ) SendMessage( hListBox,
                                                    LB_GETITEMDATA, i, 0L );
        SendMessage( hWndProcess, PM_EXITPROCESS, 0, 0L );
    }
}
}
return FALSE;
}

/*****
/* WinMain : Maint entry point of the application */
/*-----*/
/* Parameter:    Default WinMain parameters */
/* Return value: Default WinMain return value */
*****/
int WINAPI WinMain( HINSTANCE hInstance,
                   HINSTANCE hPrevInstance,
                   LPSTR      lpszCmdLine,
                   int         nCmdShow )
{
    DialogBox( NULL,
              MAKEINTRESOURCE( IDD_MASTER ),
              NULL,
              DialogProc );

    return 0;
}

```