

```

/*****
/*          O P E N S A . C          */
/*-----*/
/* Task      : Opening, saving and editing of text files. Using the      */
/*             Save dialog box as an example, the program demonstrates    */
/*             how to add and edit custom controls to the dialog box.    */
/*-----*/
/* Authors    : Michael Tischer and Bruno Jennrich                      */
/* developed on : 09/15/1995                                              */
/* last update  : 09/28/1995                                              */
/*****
#include <windows.h>
#include <shlobj.h>
#include "resource.h"

//----- Global Variables -----
char g_szFileName[ MAX_PATH ]; // Current filename
char g_szInitialDir[ MAX_PATH ]; // Initial directory for file selection
BOOL g_bHasName; // the input text already has a name
BOOL g_bChanged; // the loaded text has been changed

/*****
/* LoadFileToEdit - Loads text from file into Edit box */
/*-----*/
/* Parameter :     hDlg - Handle of dialog box */
/*             hEdit - Handle of Edit box */
/* Return value : none */
/*****
void LoadFileToEdit( HWND hDlg, HWND hEdit )
{
    OPENFILENAME ofn; // Structure for OpenFile dialog box

    ofn.lStructSize = sizeof( ofn );
    ofn.hwndOwner = hDlg; // Modal dialog box or window
    ofn.hInstance = (HINSTANCE)GetWindowLong( hDlg, GWL_HINSTANCE );
    ofn.lpstrFilter = "Text files (*.txt)\0*.txt"; // Text file filter
    ofn.lpstrCustomFilter = NULL;
    ofn.nMaxCustFilter = 0;
    ofn.nFilterIndex = 0;
    ofn.lpstrFile = g_szFileName; // Buffer for file name
    ofn.nMaxFile = sizeof( g_szFileName );
    ofn.lpstrFileTitle = 0;
    ofn.nMaxFileTitle = 0;
    ofn.lpstrInitialDir = g_szInitialDir; // Initial directory
    ofn.lpstrTitle = "Open text file"; // Dialog title
    ofn.Flags = OFN_EXPLORER | OFN_HIDEREADONLY; // Win95 Explorer style
    ofn.nFileOffset = 0;
    ofn.nFileExtension = 0;
    ofn.lpstrDefExt = "*.txt";
    ofn.lCustData = 0;
    ofn.lpfnHook = NULL;
    ofn.lpTemplateName = NULL;

    if( GetOpenFileName( &ofn ) ) // Display OpenFile dialog box
    { HANDLE hFile;

        // Get directory of filename from filename and place in --
        // g_szInitialDir. ofn.FileOffset contains the position --
        // within g_szFileName where the actual filename begins --
        // and the path ends. --
        // This initial directory will be used when the file is --
        // saved via "Save As". --
        lstrcpyn( g_szInitialDir, g_szFileName, ofn.nFileOffset );
        g_szInitialDir[ofn.nFileOffset] = '\0';

        hFile = CreateFile( g_szFileName, // open existing file reading
                           GENERIC_READ,
                           FILE_SHARE_READ,
                           NULL,
                           OPEN_EXISTING,
                           0,
                           NULL );

        if( hFile != INVALID_HANDLE_VALUE ) // File present?
        {

```

```

DWORD dwSizeLow, dwSizeHigh;

// Get length of file (max. 32K in Edit box) -----
dwSizeLow = GetFileSize( hFile, &dwSizeHigh );
if( ( !dwSizeHigh ) && ( dwSizeLow < 32767 ) )
{
    LPBYTE lpmem;

    lpmem = malloc( dwSizeLow );           // Allocate memory
    if( lpmem )
    {
        DWORD dwRead;
        // Read file and test number of read bytes -----
        if( ( ReadFile( hFile, lpmem, dwSizeLow, &dwRead, NULL ) ) &&
            ( dwRead == dwSizeLow ) )
        {
            DWORD dwStyle;

            // Pass read text to Edit box -----
            SendMessage( hEdit, WM_SETTEXT, 0, (LPARAM) lpmem );

            // Name of opened file as dialog box title -----
            SendMessage( hDlg, WM_SETTEXT, 0, (LPARAM) g_szFileName );

            dwStyle = GetWindowLong( hEdit, GWL_STYLE );
            if( ofn.Flags & OFN_READONLY )
                dwStyle |= ES_READONLY;
            else
                dwStyle &= ~ES_READONLY;
            SetWindowLong( hEdit, GWL_STYLE, dwStyle );

            g_bChanged = FALSE;           // read file was not changed
            g_bHasName = TRUE;            // read file has a name
        }
        else
            MessageBox( hDlg, "Read error", "Error", 0 );
        free( (LPVOID)lpmem );           // Release memory again
    }
    else MessageBox( hDlg, "No free memory", "Error", 0 );
}
else MessageBox( hDlg, "The file is too large!", "Error", 0 );
CloseHandle( hFile );
}
else
    MessageBox( hDlg, "Error opening file", "Error", 0 );
}
}

/*****
/* SaveDlgExtension - Save Dialog Callback function */
/*-----*/
/* Parameters:      Default dialog box parameters */
/* Return value : Default return value */
/*-----*/
/* Info : This hook will only be called when the OFN_ENABLEHOOK */
/*         flag and the ofn.lpfnHook field have been set. */
/*****

BOOL WINAPI SaveDlgExtension( HWND    hWnd,
                             UINT    wParam,
                             WPARAM  wp,
                             LPARAM  lp )
{
    static BOOL *pbAddToRecentDocs;           // Address of user variables
                                           // in static variables

    switch( wParam )
    {
        case WM_INITDIALOG:
            // As lParam, this function expects the address of a BOOL --
            // variable that records whether the file is to be added to --
            // the document list of the Start menu during saving. --
            pbAddToRecentDocs = (BOOL *)lp;

            // Set state of checkbox according to *pbAddToRecentDocs --
            SendMessage( GetDlgItem( hWnd, IDC_ADDTORECENTDOCS ),
                          BM_SETCHECK,
                          *pbAddToRecentDocs,

```

```

        0 );
break;
case WM_COMMAND:
    switch( LOWORD( wp ) )
    {
        case IDC_ADDTORECENTDOCS:
            // Save current state of checkbox -----
            *pbAddToRecentDocs = IsDlgButtonChecked( hWnd,
                                                    LOWORD( wp ) );
            break;
    }
break;
}
return FALSE;
}

/*****
/* SaveEditToFile - Saves text from Edit box to file */
/*-----*/
/* Parameters:      hDlg      - Handle of dialog box */
/*                  hEdit     - Handle of Edit box */
/*                  bSaveAs   - Save text under a different name? */
/* Return value : TRUE  - Save OK */
/*                  FALSE - Error */
/*-----*/
/* Info : The "Save As" dialog box gets a checkbox added */
/*          to it. To achieve this, the ID of a dialog resource */
/*          is specified in ofn.lpTemplateName. This checkbox is placed */
/*          under the default controls of the Open/Save dialog box. */
/*****/
BOOL SaveEditToFile( HWND hDlg, HWND hEdit, BOOL bSaveAs )
{
    BOOL bRetVal = FALSE;
    BOOL bAddToRecentDocs = FALSE;
    LPBYTE lpmem;
    DWORD dwSize;
    HANDLE hFile;

    // Call Save dialog box when "Save As..." was called or text -
    // was entered that hasn't yet been saved -
    if( ( !g_bHasName ) || ( bSaveAs ) )
    {
        OPENFILENAME ofn; // Dialog box structure

        ofn.lStructSize = sizeof( ofn );
        ofn.hwndOwner = hDlg;
        ofn.hInstance = (HINSTANCE)GetWindowLong( hDlg, GWL_HINSTANCE );
        ofn.lpstrFilter = "Text files (*.txt)\0*.txt";
        ofn.lpstrCustomFilter = NULL;
        ofn.nMaxCustFilter = 0;
        ofn.nFilterIndex = 0;
        ofn.lpstrFile = g_szFileName; // Specify current filename
        ofn.nMaxFile = sizeof( g_szFileName );
        ofn.lpstrFileName = 0;
        ofn.nMaxFileName = 0;
        ofn.lpstrInitialDir = g_szInitialDir;
        if( bSaveAs )
            ofn.lpstrTitle = "Save text file as...";
        else
            ofn.lpstrTitle = "Save text file";
        ofn.Flags = OFN_EXPLORER |
                    OFN_ENABLETEMPLATE | // Expand dialog box
                    OFN_ENABLEHOOK | // Enable callback
                    OFN_HIDEREADONLY; // Hide "Read only"
        ofn.nFileOffset = 0;
        ofn.nFileExtension = 0;
        ofn.lpstrDefExt = "*.txt";
        ofn.lCustData = (long)&bAddToRecentDocs;
        ofn.lpfnHook = SaveDlgExtension; // Address of callback
        ofn.lpTemplateName = MAKEINTRESOURCE( IDD_SAVEEXTENSION );

        if( !GetSaveFileName( &ofn ) ) return FALSE; // Show Save dialog box

        // If the user confirmed the dialog box with OK, the current ---
        // Save directory is determined in order to save the file ---
    }
}

```



```

        free( lpmem );                                // Release memory!
        return bRetVal;
    }
    else return TRUE;
}

/*****
/* TestSave - Checks whether current document needs to be saved when
/*              opening a document.
/*-----*/
/* Parameters:   hWnd    - Handle of dialog box, so that a
/*                  message appears in the modal window.
/* Return value : TRUE   - Text needs to be saved
/*                  FALSE - Discard text
*****/
BOOL TestSave( HWND hWnd )
{
    if( g_bChanged )                                // Prompt only if text has been changed
        if( MessageBox( hWnd,
            "Do you want to discard the current file?",
            "Question",
            MB_YESNO ) == IDNO ) return TRUE;

    return FALSE;
}

/*****
/* TestExit - Checks whether the changed text needs to be saved
/*              before exiting the program, and then performs the
/*              save.
/*-----*/
/* Parameters:   hWnd    - Handle of dialog box, so that a
/*                  message appears in the modal window.
/* Return value : TRUE   - Program can be exited
/*                  FALSE - Do not exit program
*****/
BOOL TestExit( HWND hDlg, HWND hEdit )
{
    int iRetVal;
    if( g_bChanged )
    {
        iRetVal = MessageBox( hDlg,
            "Do you want to save the document before exiting?",
            "Warning",
            MB_YESNOCANCEL );

        if( iRetVal == IDYES )
            return SaveEditToFile( hDlg, hEdit, FALSE );
        return ( iRetVal == IDNO );
    }
    return TRUE;
}

/*****
/* SimpleEditDlgProc - Editor dialog function
/*-----*/
/* Parameters:   Default dialog parameters
/* Return value : Default return value
/*-----*/
/* Info : This function edits the menu commands and resizes
/*          the Edit box to fit correctly.
*****/
BOOL WINAPI SimpleEditDlgProc( HWND hWnd,
                               UINT wMsg,
                               WPARAM wp,
                               LPARAM lp )
{
    switch( wMsg )
    {
        case WM_INITDIALOG:
            // Set Text1.txt as dialog title
            SendMessage( hWnd, WM_SETTEXT, 0, (LPARAM)g_szFileName );
            break;
        case WM_SIZE:
            // Resize the Edit box
            MoveWindow( GetDlgItem( hWnd, IDC_EDIT ),
                0, 0,
                LOWORD( lp ),
                HIWORD( lp ), TRUE );
    }
}

```

```

break;
case WM_COMMAND:                // Process commands triggered by user
    switch( LOWORD( wp ) )
    {
        case IDCANCEL:          // Close button pressed
            // Save changed text if necessary -----
            if( !TestExit( hWnd, GetDlgItem( hWnd, IDC_EDIT ) ) )
                break;

            // Exit dialog box, thus ending application -----
            EndDialog( hWnd, 0 );
            break;

        case IDC_EDIT:          // Register changed in Edit box
            if( HIWORD( wp ) == EN_CHANGE )
            {
                if( !g_bChanged )    // Expand title bar by " *"
                {
                    char szTitle[ MAX_PATH + 3 ];

                    SendMessage( hWnd,          // Get current title bar
                                WM_GETTEXT,
                                MAX_PATH,
                                (LPARAM) szTitle );
                    lstrcat( szTitle, " *");          // Add " *"
                    SendMessage( hWnd,          // and display new title bar
                                WM_SETTEXT,
                                0,
                                (LPARAM) szTitle );
                    g_bChanged = TRUE;
                }
            }
            break;
        case ID_FILE_OPEN1:      // User wants to load new text

            if( TestSave( hWnd ) )    // Save current text?
                SaveEditToFile( hWnd,
                                GetDlgItem( hWnd, IDC_EDIT ),
                                FALSE );

            // Load new text -----
            LoadFileToEdit( hWnd, GetDlgItem( hWnd, IDC_EDIT ) );
            break;
        case ID_FILE_SAVE1:
            // Save text under current name. However, if the input
            // text doesn't have a name, you will be prompted to
            // enter one.
            SaveEditToFile( hWnd, GetDlgItem( hWnd, IDC_EDIT ), FALSE );
            break;
        case ID_FILE_SAVEAS:
            // Save text as different name -----
            SaveEditToFile( hWnd, GetDlgItem( hWnd, IDC_EDIT ), TRUE );
            break;
    }
    break;
}
return FALSE;
}

/*****
/* WinMain - Start function */
/*-----*/
/* Parameters:    Default parameters */
/* Return value : Default return value */
*****/
int WINAPI WinMain( HINSTANCE hInst,
                   HINSTANCE hPrev,
                   LPSTR lpCmdLine,
                   int nCmdShow )
{
    g_bChanged = FALSE;          // Current text has not been changed
    g_bHasName = FALSE;          // Current text doesn't have a name

    // Display current working directory in File dialog boxes -----
    GetCurrentDirectory( MAX_PATH, g_szInitialDir );

```

```
lstrcpy( g_szFileName, "Text1.txt" );           // Set default filename

// Display Editor -----
DialogBox( hInst,
           MAKEINTRESOURCE( IDD_SIMPLEEDIT ),
           NULL,
           SimpleEditDlgProc );
return 0;
}
```