[**GetOpenFileName**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646927(v=vs.85).aspx)

TCHAR szFile[MAX\_PATH] = {0};

OPENFILENAME ofn;

memset(&ofn, 0x00 ,sizeof(ofn));

ofn.lStructSize = sizeof(ofn);

ofn.lpstrFilter = "All Files\0\*.\*\0\0";

ofn.lpstrFileTitle = "打開文件";

ofn.nMaxFile = MAX\_PATH;

ofn.lpstrFile = szFile;

if(GetOpenFileName(&ofn))

{

SetDlgItemText(IDC\_EDIT1,szFile);

}

typedef struct tagOFN {

DWORD         lStructSize;

HWND          hwndOwner;

HINSTANCE     hInstance;

LPCTSTR       lpstrFilter;

LPTSTR        lpstrCustomFilter;

DWORD         nMaxCustFilter;

DWORD         nFilterIndex;

LPTSTR        lpstrFile;

DWORD         nMaxFile;

LPTSTR        lpstrFileTitle;

DWORD         nMaxFileTitle;

LPCTSTR       lpstrInitialDir;

LPCTSTR       lpstrTitle;

DWORD         Flags;

WORD          nFileOffset;

WORD          nFileExtension;

LPCTSTR       lpstrDefExt;

LPARAM        lCustData;

LPOFNHOOKPROC lpfnHook;

LPCTSTR       lpTemplateName;

#if (\_WIN32\_WINNT >= 0x0500)

void          \*pvReserved;

DWORD         dwReserved;

DWORD         FlagsEx;

#endif

} OPENFILENAME, \*LPOPENFILENAME;

**Members**

**lStructSize**

Type: **DWORD**

The length, in bytes, of the structure. Use sizeof (OPENFILENAME) for this parameter.

**hwndOwner**

Type: **HWND**

A handle to the window that owns the dialog box. This member can be any valid window handle, or it can be **NULL** if the dialog box has no owner.

**hInstance**

Type: **HINSTANCE**

If the **OFN\_ENABLETEMPLATEHANDLE** flag is set in the **Flags** member, **hInstance** is a handle to a memory object containing a dialog box template. If the **OFN\_ENABLETEMPLATE** flag is set, **hInstance** is a handle to a module that contains a dialog box template named by the **lpTemplateName** member. If neither flag is set, this member is ignored. If the **OFN\_EXPLORER** flag is set, the system uses the specified template to create a dialog box that is a child of the default Explorer-style dialog box. If the **OFN\_EXPLORER** flag is not set, the system uses the template to create an old-style dialog box that replaces the default dialog box.

**lpstrFilter**

Type: **LPCTSTR**

A buffer containing pairs of null-terminated filter strings. The last string in the buffer must be terminated by two **NULL** characters.

The first string in each pair is a display string that describes the filter (for example, "Text Files"), and the second string specifies the filter pattern (for example, "\*.TXT"). To specify multiple filter patterns for a single display string, use a semicolon to separate the patterns (for example, "\*.TXT;\*.DOC;\*.BAK"). A pattern string can be a combination of valid file name characters and the asterisk (\*) wildcard character. Do not include spaces in the pattern string.

The system does not change the order of the filters. It displays them in the **File Types** combo box in the order specified in **lpstrFilter**.

If **lpstrFilter** is **NULL**, the dialog box does not display any filters.

In the case of a shortcut, if no filter is set, [**GetOpenFileName**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646927(v=vs.85).aspx) and [**GetSaveFileName**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646928(v=vs.85).aspx) retrieve the name of the .lnk file, not its target. This behavior is the same as setting the **OFN\_NODEREFERENCELINKS** flag in the **Flags** member. To retrieve a shortcut's target without filtering, use the string "All Files\0\*.\*\0\0".

**lpstrCustomFilter**

Type: **LPTSTR**

A static buffer that contains a pair of null-terminated filter strings for preserving the filter pattern chosen by the user. The first string is your display string that describes the custom filter, and the second string is the filter pattern selected by the user. The first time your application creates the dialog box, you specify the first string, which can be any nonempty string. When the user selects a file, the dialog box copies the current filter pattern to the second string. The preserved filter pattern can be one of the patterns specified in the **lpstrFilter** buffer, or it can be a filter pattern typed by the user. The system uses the strings to initialize the user-defined file filter the next time the dialog box is created. If the **nFilterIndex** member is zero, the dialog box uses the custom filter.

If this member is **NULL**, the dialog box does not preserve user-defined filter patterns.

If this member is not **NULL**, the value of the **nMaxCustFilter** member must specify the size, in characters, of the **lpstrCustomFilter** buffer.

**nMaxCustFilter**

Type: **DWORD**

The size, in characters, of the buffer identified by **lpstrCustomFilter**. This buffer should be at least 40 characters long. This member is ignored if **lpstrCustomFilter** is **NULL** or points to a **NULL** string.

**nFilterIndex**

Type: **DWORD**

The index of the currently selected filter in the **File Types** control. The buffer pointed to by **lpstrFilter** contains pairs of strings that define the filters. The first pair of strings has an index value of 1, the second pair 2, and so on. An index of zero indicates the custom filter specified by **lpstrCustomFilter**. You can specify an index on input to indicate the initial filter description and filter pattern for the dialog box. When the user selects a file, **nFilterIndex** returns the index of the currently displayed filter. If **nFilterIndex** is zero and **lpstrCustomFilter** is **NULL**, the system uses the first filter in the **lpstrFilter** buffer. If all three members are zero or **NULL**, the system does not use any filters and does not show any files in the file list control of the dialog box.

**lpstrFile**

Type: **LPTSTR**

The file name used to initialize the **File Name** edit control. The first character of this buffer must be **NULL** if initialization is not necessary. When the [**GetOpenFileName**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646927(v=vs.85).aspx) or [**GetSaveFileName**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646928(v=vs.85).aspx) function returns successfully, this buffer contains the drive designator, path, file name, and extension of the selected file.

If the **OFN\_ALLOWMULTISELECT** flag is set and the user selects multiple files, the buffer contains the current directory followed by the file names of the selected files. For Explorer-style dialog boxes, the directory and file name strings are **NULL** separated, with an extra **NULL** character after the last file name. For old-style dialog boxes, the strings are space separated and the function uses short file names for file names with spaces. You can use the [**FindFirstFile**](http://msdn.microsoft.com/en-us/library/windows/desktop/aa364418(v=vs.85).aspx) function to convert between long and short file names. If the user selects only one file, the **lpstrFile** string does not have a separator between the path and file name.

If the buffer is too small, the function returns **FALSE** and the [**CommDlgExtendedError**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646916(v=vs.85).aspx) function returns **FNERR\_BUFFERTOOSMALL**. In this case, the first two bytes of the **lpstrFile** buffer contain the required size, in bytes or characters.

**nMaxFile**

Type: **DWORD**

The size, in characters, of the buffer pointed to by **lpstrFile**. The buffer must be large enough to store the path and file name string or strings, including the terminating **NULL** character. The [**GetOpenFileName**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646927(v=vs.85).aspx) and [**GetSaveFileName**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646928(v=vs.85).aspx) functions return **FALSE** if the buffer is too small to contain the file information. The buffer should be at least 256 characters long.

**lpstrFileTitle**

Type: **LPTSTR**

The file name and extension (without path information) of the selected file. This member can be **NULL**.

**nMaxFileTitle**

Type: **DWORD**

The size, in characters, of the buffer pointed to by **lpstrFileTitle**. This member is ignored if **lpstrFileTitle** is **NULL**.

**lpstrInitialDir**

Type: **LPCTSTR**

The initial directory. The algorithm for selecting the initial directory varies on different platforms.

**Windows 7:**

1. If **lpstrInitialDir** has the same value as was passed the first time the application used an **Open** or **Save As** dialog box, the path most recently selected by the user is used as the initial directory.
2. Otherwise, if **lpstrFile** contains a path, that path is the initial directory.
3. Otherwise, if **lpstrInitialDir** is not **NULL**, it specifies the initial directory.
4. If **lpstrInitialDir** is **NULL** and the current directory contains any files of the specified filter types, the initial directory is the current directory.
5. Otherwise, the initial directory is the personal files directory of the current user.
6. Otherwise, the initial directory is the Desktop folder.

**Windows 2000/XP/Vista:**

1. If **lpstrFile** contains a path, that path is the initial directory.
2. Otherwise, **lpstrInitialDir** specifies the initial directory.
3. Otherwise, if the application has used an **Open** or **Save As** dialog box in the past, the path most recently used is selected as the initial directory. However, if an application is not run for a long time, its saved selected path is discarded.
4. If **lpstrInitialDir** is **NULL** and the current directory contains any files of the specified filter types, the initial directory is the current directory.
5. Otherwise, the initial directory is the personal files directory of the current user.
6. Otherwise, the initial directory is the Desktop folder.

**lpstrTitle**

Type: **LPCTSTR**

A string to be placed in the title bar of the dialog box. If this member is **NULL**, the system uses the default title (that is, **Save As** or **Open**).

**Flags**

Type: **DWORD**

A set of bit flags you can use to initialize the dialog box. When the dialog box returns, it sets these flags to indicate the user's input. This member can be a combination of the following flags.

|  |  |
| --- | --- |
| **Value** | **Meaning** |
| **OFN\_ALLOWMULTISELECT**  0x00000200 | The **File Name** list box allows multiple selections. If you also set the **OFN\_EXPLORER** flag, the dialog box uses the Explorer-style user interface; otherwise, it uses the old-style user interface.  If the user selects more than one file, the **lpstrFile** buffer returns the path to the current directory followed by the file names of the selected files. The **nFileOffset** member is the offset, in bytes or characters, to the first file name, and the **nFileExtension** member is not used. For Explorer-style dialog boxes, the directory and file name strings are **NULL** separated, with an extra **NULL** character after the last file name. This format enables the Explorer-style dialog boxes to return long file names that include spaces. For old-style dialog boxes, the directory and file name strings are separated by spaces and the function uses short file names for file names with spaces. You can use the [**FindFirstFile**](http://msdn.microsoft.com/en-us/library/windows/desktop/aa364418(v=vs.85).aspx) function to convert between long and short file names.  If you specify a custom template for an old-style dialog box, the definition of the **File Name** list box must contain the **LBS\_EXTENDEDSEL** value. |
| **OFN\_CREATEPROMPT**  0x00002000 | If the user specifies a file that does not exist, this flag causes the dialog box to prompt the user for permission to create the file. If the user chooses to create the file, the dialog box closes and the function returns the specified name; otherwise, the dialog box remains open. If you use this flag with the **OFN\_ALLOWMULTISELECT** flag, the dialog box allows the user to specify only one nonexistent file. |
| **OFN\_DONTADDTORECENT**  0x02000000 | Prevents the system from adding a link to the selected file in the file system directory that contains the user's most recently used documents. To retrieve the location of this directory, call the [**SHGetSpecialFolderLocation**](http://msdn.microsoft.com/en-us/library/windows/desktop/bb762203(v=vs.85).aspx) function with the **CSIDL\_RECENT** flag. |
| **OFN\_ENABLEHOOK**  0x00000020 | Enables the hook function specified in the **lpfnHook** member. |
| **OFN\_ENABLEINCLUDENOTIFY**  0x00400000 | Causes the dialog box to send [**CDN\_INCLUDEITEM**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646862(v=vs.85).aspx) notification messages to your [*OFNHookProc*](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646931(v=vs.85).aspx) hook procedure when the user opens a folder. The dialog box sends a notification for each item in the newly opened folder. These messages enable you to control which items the dialog box displays in the folder's item list. |
| **OFN\_ENABLESIZING**  0x00800000 | Enables the Explorer-style dialog box to be resized using either the mouse or the keyboard. By default, the Explorer-style **Open** and **Save As** dialog boxes allow the dialog box to be resized regardless of whether this flag is set. This flag is necessary only if you provide a hook procedure or custom template. The old-style dialog box does not permit resizing. |
| **OFN\_ENABLETEMPLATE**  0x00000040 | The **lpTemplateName** member is a pointer to the name of a dialog template resource in the module identified by the **hInstance** member. If the **OFN\_EXPLORER** flag is set, the system uses the specified template to create a dialog box that is a child of the default Explorer-style dialog box. If the **OFN\_EXPLORER** flag is not set, the system uses the template to create an old-style dialog box that replaces the default dialog box. |
| **OFN\_ENABLETEMPLATEHANDLE**  0x00000080 | The **hInstance** member identifies a data block that contains a preloaded dialog box template. The system ignores **lpTemplateName** if this flag is specified. If the **OFN\_EXPLORER** flag is set, the system uses the specified template to create a dialog box that is a child of the default Explorer-style dialog box. If the **OFN\_EXPLORER** flag is not set, the system uses the template to create an old-style dialog box that replaces the default dialog box. |
| **OFN\_EXPLORER**  0x00080000 | Indicates that any customizations made to the **Open** or **Save As** dialog box use the Explorer-style customization methods. For more information, see [Explorer-Style Hook Procedures](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646960(v=vs.85).aspx#_win32_Explorer_Style_Hook_Procedures) and [Explorer-Style Custom Templates](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646960(v=vs.85).aspx#_win32_Explorer_Style_Custom_Templates).  By default, the **Open** and **Save As** dialog boxes use the Explorer-style user interface regardless of whether this flag is set. This flag is necessary only if you provide a hook procedure or custom template, or set the **OFN\_ALLOWMULTISELECT** flag.  If you want the old-style user interface, omit the **OFN\_EXPLORER** flag and provide a replacement old-style template or hook procedure. If you want the old style but do not need a custom template or hook procedure, simply provide a hook procedure that always returns **FALSE**. |
| **OFN\_EXTENSIONDIFFERENT**  0x00000400 | The user typed a file name extension that differs from the extension specified by **lpstrDefExt**. The function does not use this flag if **lpstrDefExt** is **NULL**. |
| **OFN\_FILEMUSTEXIST**  0x00001000 | The user can type only names of existing files in the **File Name** entry field. If this flag is specified and the user enters an invalid name, the dialog box procedure displays a warning in a message box. If this flag is specified, the **OFN\_PATHMUSTEXIST** flag is also used. This flag can be used in an **Open** dialog box. It cannot be used with a **Save As** dialog box. |
| **OFN\_FORCESHOWHIDDEN**  0x10000000 | Forces the showing of system and hidden files, thus overriding the user setting to show or not show hidden files. However, a file that is marked both system and hidden is not shown. |
| **OFN\_HIDEREADONLY**  0x00000004 | Hides the **Read Only** check box. |
| **OFN\_LONGNAMES**  0x00200000 | For old-style dialog boxes, this flag causes the dialog box to use long file names. If this flag is not specified, or if the **OFN\_ALLOWMULTISELECT** flag is also set, old-style dialog boxes use short file names (8.3 format) for file names with spaces. Explorer-style dialog boxes ignore this flag and always display long file names. |
| **OFN\_NOCHANGEDIR**  0x00000008 | Restores the current directory to its original value if the user changed the directory while searching for files.  This flag is ineffective for [**GetOpenFileName**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646927(v=vs.85).aspx). |
| **OFN\_NODEREFERENCELINKS**  0x00100000 | Directs the dialog box to return the path and file name of the selected shortcut (.LNK) file. If this value is not specified, the dialog box returns the path and file name of the file referenced by the shortcut. |
| **OFN\_NOLONGNAMES**  0x00040000 | For old-style dialog boxes, this flag causes the dialog box to use short file names (8.3 format). Explorer-style dialog boxes ignore this flag and always display long file names. |
| **OFN\_NONETWORKBUTTON**  0x00020000 | Hides and disables the **Network** button. |
| **OFN\_NOREADONLYRETURN**  0x00008000 | The returned file does not have the **Read Only** check box selected and is not in a write-protected directory. |
| **OFN\_NOTESTFILECREATE**  0x00010000 | The file is not created before the dialog box is closed. This flag should be specified if the application saves the file on a create-nonmodify network share. When an application specifies this flag, the library does not check for write protection, a full disk, an open drive door, or network protection. Applications using this flag must perform file operations carefully, because a file cannot be reopened once it is closed. |
| **OFN\_NOVALIDATE**  0x00000100 | The common dialog boxes allow invalid characters in the returned file name. Typically, the calling application uses a hook procedure that checks the file name by using the [**FILEOKSTRING**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646870(v=vs.85).aspx) message. If the text box in the edit control is empty or contains nothing but spaces, the lists of files and directories are updated. If the text box in the edit control contains anything else, **nFileOffset** and **nFileExtension** are set to values generated by parsing the text. No default extension is added to the text, nor is text copied to the buffer specified by **lpstrFileTitle**. If the value specified by **nFileOffset** is less than zero, the file name is invalid. Otherwise, the file name is valid, and **nFileExtension** and **nFileOffset** can be used as if the **OFN\_NOVALIDATE** flag had not been specified. |
| **OFN\_OVERWRITEPROMPT**  0x00000002 | Causes the **Save As** dialog box to generate a message box if the selected file already exists. The user must confirm whether to overwrite the file. |
| **OFN\_PATHMUSTEXIST**  0x00000800 | The user can type only valid paths and file names. If this flag is used and the user types an invalid path and file name in the **File Name** entry field, the dialog box function displays a warning in a message box. |
| **OFN\_READONLY**  0x00000001 | Causes the **Read Only** check box to be selected initially when the dialog box is created. This flag indicates the state of the **Read Only** check box when the dialog box is closed. |
| **OFN\_SHAREAWARE**  0x00004000 | Specifies that if a call to the [**OpenFile**](http://msdn.microsoft.com/en-us/library/windows/desktop/aa365430(v=vs.85).aspx) function fails because of a network sharing violation, the error is ignored and the dialog box returns the selected file name. If this flag is not set, the dialog box notifies your hook procedure when a network sharing violation occurs for the file name specified by the user. If you set the **OFN\_EXPLORER** flag, the dialog box sends the [**CDN\_SHAREVIOLATION**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646866(v=vs.85).aspx) message to the hook procedure. If you do not set **OFN\_EXPLORER**, the dialog box sends the [**SHAREVISTRING**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646878(v=vs.85).aspx) registered message to the hook procedure. |
| **OFN\_SHOWHELP**  0x00000010 | Causes the dialog box to display the **Help** button. The **hwndOwner** member must specify the window to receive the [**HELPMSGSTRING**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646874(v=vs.85).aspx) registered messages that the dialog box sends when the user clicks the **Help** button. An Explorer-style dialog box sends a [**CDN\_HELP**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646860(v=vs.85).aspx) notification message to your hook procedure when the user clicks the **Help** button. |

**nFileOffset**

Type: **WORD**

The zero-based offset, in characters, from the beginning of the path to the file name in the string pointed to by **lpstrFile**. For the ANSI version, this is the number of bytes; for the Unicode version, this is the number of characters. For example, if **lpstrFile** points to the following string, "c:\dir1\dir2\file.ext", this member contains the value 13 to indicate the offset of the "file.ext" string. If the user selects more than one file, **nFileOffset** is the offset to the first file name.

**nFileExtension**

Type: **WORD**

The zero-based offset, in characters, from the beginning of the path to the file name extension in the string pointed to by **lpstrFile**. For the ANSI version, this is the number of bytes; for the Unicode version, this is the number of characters. Usually the file name extension is the substring which follows the last occurrence of the dot (".") character. For example, txt is the extension of the filename readme.txt, html the extension of readme.txt.html. Therefore, if **lpstrFile** points to the string "c:\dir1\dir2\readme.txt", this member contains the value 20. If **lpstrFile** points to the string "c:\dir1\dir2\readme.txt.html", this member contains the value 24. If **lpstrFile** points to the string "c:\dir1\dir2\readme.txt.html.", this member contains the value 29. If **lpstrFile** points to a string that does not contain any "." character such as "c:\dir1\dir2\readme", this member contains zero.

**lpstrDefExt**

Type: **LPCTSTR**

The default extension. [**GetOpenFileName**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646927(v=vs.85).aspx) and [**GetSaveFileName**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646928(v=vs.85).aspx) append this extension to the file name if the user fails to type an extension. This string can be any length, but only the first three characters are appended. The string should not contain a period (.). If this member is **NULL** and the user fails to type an extension, no extension is appended.

**lCustData**

Type: **LPARAM**

Application-defined data that the system passes to the hook procedure identified by the **lpfnHook** member. When the system sends the [**WM\_INITDIALOG**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms645428(v=vs.85).aspx) message to the hook procedure, the message's *lParam* parameter is a pointer to the **OPENFILENAME** structure specified when the dialog box was created. The hook procedure can use this pointer to get the **lCustData** value.

**lpfnHook**

Type: **LPOFNHOOKPROC**

A pointer to a hook procedure. This member is ignored unless the **Flags** member includes the **OFN\_ENABLEHOOK** flag.

If the **OFN\_EXPLORER** flag is not set in the **Flags** member, **lpfnHook** is a pointer to an [*OFNHookProcOldStyle*](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646932(v=vs.85).aspx) hook procedure that receives messages intended for the dialog box. The hook procedure returns **FALSE** to pass a message to the default dialog box procedure or **TRUE** to discard the message.

If **OFN\_EXPLORER** is set, **lpfnHook** is a pointer to an [*OFNHookProc*](http://msdn.microsoft.com/en-us/library/windows/desktop/ms646931(v=vs.85).aspx) hook procedure. The hook procedure receives notification messages sent from the dialog box. The hook procedure also receives messages for any additional controls that you defined by specifying a child dialog template. The hook procedure does not receive messages intended for the standard controls of the default dialog box.

**lpTemplateName**

Type: **LPCTSTR**

The name of the dialog template resource in the module identified by the **hInstance** member. For numbered dialog box resources, this can be a value returned by the [**MAKEINTRESOURCE**](http://msdn.microsoft.com/en-us/library/windows/desktop/ms648029(v=vs.85).aspx) macro. This member is ignored unless the **OFN\_ENABLETEMPLATE** flag is set in the **Flags** member. If the **OFN\_EXPLORER** flag is set, the system uses the specified template to create a dialog box that is a child of the default Explorer-style dialog box. If the **OFN\_EXPLORER** flag is not set, the system uses the template to create an old-style dialog box that replaces the default dialog box.

**pvReserved**

Type: **void\***

This member is reserved.

**dwReserved**

Type: **DWORD**

This member is reserved.

**FlagsEx**

Type: **DWORD**

A set of bit flags you can use to initialize the dialog box. Currently, this member can be zero or the following flag.

|  |  |
| --- | --- |
| **Value** | **Meaning** |
| **OFN\_EX\_NOPLACESBAR**  0x00000001 | If this flag is set, the places bar is not displayed. If this flag is not set, Explorer-style dialog boxes include a places bar containing icons for commonly-used folders, such |