WinX Help Contents

Welcome to the Help system for the *WinX: Windows Extensions* application. Click on the green underlined words to jump to a specific topic. Click the "Back" button to return. Use the ">>" and "<<" buttons to read this document like a book. Click the mouse on the following words--Getting Started--to get started.

Overview

Main Window Controls

In Case of Trouble

Getting Started

WinX makes Windows easier to use

The main feature of WinX is to allow any window to be moved and used without it first popping to the front. The user, not Windows, is in control of window depth, or Z position. This makes window arrangements on the desktop easier to maintain, and side-by-side use of applications more effective.

WinX features are easy to use

Window Z position is controlled by positioning the mouse cursor on a **window border** or **title bar**. Clicking the **mouse left** button pops the window to the front. Pressing the **mouse right** button pops up a menu of commands which move the window backwards and forwards. Equivalent keyboard **command keys** are also provided.

WinX works *with* Windows

WinX works quietly in the background with Windows to provide user interface features commonly found on workstations. These features have been adapted to work with--not replace--the user interface features of Windows. The use of its features is so transparent that WinX is usually left minimized on the desktop--there is rarely any need to use its main window controls.

Windows vs. WinX

The following table compares the user interface characteristics of Windows against the full set of features provided by WinX. See Why Use It for a more detailed discussion.

Windows	WinX
Windows always pop to the front.	Windows maintain their Z (depth) position when sized, moved or used.
User not in control of Z position.	Popup menu and command keys move windows forward and back.
Drag title bar to move window.	Drag window border or title bar to move window.
Click on window to activate it.	Window under mouse cursor automatically activates.
Click outside of menu to close it.	Menu always closes when mouse button is released.
User not in control of features.	All user interface features can be enabled and disabled.

Why Use It

WinX provides many benefits to the user, not the least of which are less wasted time and aggravation from shuffling windows around on the desktop and, as a result, more effective side-by-side use of applications. The following briefly describes the features and benefits of using the full capabilities of WinX.

Activated windows maintain their Z (depth) position:

In Windows, in order to size, move or use a window it must be activated. When a window is activated it is also brought to the top of the Z (depth) order on the desktop. For busy desktops with multiple open applications this can mean a lot of wasted time pushing large windows out of the way or making desktop windows smaller and scrolling them.

With WinX, any part of an application that is visible can be used without its window first popping to the front (see Maintain Z Extension). For the same reason, an individual window can be moved or sized without disturbing the arrangement of other windows on the desktop.

The user, not Windows, is in control of window Z position:

In Windows, there is little user control of window Z position. Windows can only be brought to the front, by activating them. Finding a hidden window requires moving, sizing, or minimizing overlying windows, or using the Program Manager Task List. Rearranging the desktop each time is frustrating, and selecting the desired window from a list of task names can be error prone.

With WinX, a menu of Z movement commands (front, forward, backward, back) can be popped up on any window border or title bar with the right mouse button. Windows can be easily moved forward and backward until the desired window or portion of a window is exposed, with minimal disturbance of the desktop window arrangement (see Accelerator Menu Extension).

Windows can be moved as well as sized with the border:

In Windows, a window can only be moved by dragging its title bar. If the title bar is covered the window must be first brought to the front and moved. Windows has no provisions for pushing it back into its original Z position. Also, windows can not be moved off the top edge of the display, only the bottom and sides.

With WinX, a window can be moved by dragging the title bar or any border side, and sized by dragging any border corner (see Size/Move Border Extension). This is especially important since WinX allows windows to be moved without first bringing them to the front (see Maintain Z Extension).

The window under the mouse is automatically activated:

In Windows, the user must click on a window to activate it. Inadvertent window movement, scrolling, and button pushes can result.

With WinX, windows are automatically activated and deactivated as the mouse passes over them (see Maintain Z Extension). The window that the mouse is over is always ready for action. An extra click of the mouse is not needed, and the window maintains its Z position (see Maintain Z Extension).

Menus always close when the mouse button is released:

In Windows, menus generally stay up after they drop down or pop up. This is often a useful feature for the novice, but can become annoying as one gains greater facility with the mouse. It can make using the mouse on menus seem slow.

With WinX, menus are always closed when the mouse button is released (see Close Menus Extension). This makes for a "snappier" user interface by avoiding extra mouse clicks, and helps to avoid accidental menu selections and spurious mouse actions.

Extensions can be individually enabled:

Since users often have strong preferences as to the "feel" of a user interface, the user can individually enable or disable each WinX extension option. Many extensions even have controllable sub-options.

Besides user customization, individual control of options addresses the occasional problem of conflicts between WinX and another application. Such conflicts can be simply resolved by disabling the offending WinX option or sub-option (see Solving Problems).

About WinX

Name, Version, Copyright

WinX: Windows Extensions

Version 1.0

Description

WinX extends Windows 3.1 to provide workstation-like windowing features. Any Window on the desktop can be moved and used without it first popping to the front. This makes window arrangement and side-by-side use of applications easier. (See also Getting Started and Why Use It.)

Distribution, Questions, Comments

WinX is distributed as shareware. If you find this product useful please register it by sending the product name, version, and $25 to its author:

Jon Barrilleaux

3800 Lake Shore Ave.

Oakland, CA 94610

Questions and comments can be sent via e-mail to:

71461.432@compuserve.com

Disclaimer and Trademarks

Disclaimer

Jon Barrilleaux, the author, hereby disclaims all warranties relating to this software, whether expressed or implied, including without limitation any implied warranties of merchantability or fitness for a particular purpose. The author will not be liable for any special, incidental, consequential, indirect or similar damages due to loss of data or any other reason even if the author or an agent of the author has been advised of the possibility of such damages. The person using the software bears all risk as to the quality and performance of the software.

Trademarks

For ease of reading this document, Windows refers to Microsoft Windows. Microsoft is a registered trademark and Windows is a trademark of Microsoft Corporation.

Windows Extensions (Group)

The Windows Extensions Group in the WinX main window provides a set of *options* (checkboxes) and *sub-options* (indented checkboxes) which enable and disable extensions--added features--to the Windows user interface. Disabling an option also disables the functionality of its sub-options. Its sub-options, however, can still be checked and unchecked.

Each extension is implemented as an independent body of software which is installed in Windows. When an extension is disabled that piece of software is removed from Windows and no longer executes. Other extensions which are enabled, however, remain fully active. This is important in situations where options need to be disabled in order to avoid occasional conflicts with other applications (see Solving Problems).

See Also

Refer to the following topics and sub-topics for a description of each Windows Extension, its effect on Windows, and its option and sub-option controls.

Accelerator Menu Extension

The Accelerator Menu Extension provides a set of Accelerator Menu Commands which allow a window to be moved forward and backward in Z position (depth), or to be changed to "Always On Top" status. The commands can be used regardless of the window's show state (minimized, normalized, or maximized).

Commands are accessed in the popup Accelerator Menu by pressing the mouse right button while the mouse cursor is on a window border or title bar. Some commands also can be accessed through keyboard command keys, again while the mouse cursor is on the window border or title bar.

As a convenience, window show state commands such as Minimize and Maximize are also included in the Accelerator Menu. This makes common system menu commands available anywhere on the window border, not just through the window's system menu box (the box left of the window title bar).

MDI Child Windows

Main windows and Multiple Document Interface (MDI) Child windows exist in separate Z orders. An MDI Child window can only be moved as far back as its main window, and only as far front as its siblings. Other main windows can not be moved in-between an MDI Child window and its main window.

For example, the File Manager window (a main window) can not be moved between the Program Manager window (an MDI main window) and any of its "group" windows (its MDI Child windows).

Modal Dialogs

As described in Automatic Activation under the **Disabled Windows** section, when a modal dialog is open the windows that own it, directly or indirectly, will be disabled. Windows blocks all mouse actions to disabled windows, which prevents an Accelerator Menu from being popped up on its border. Keyboard commands, however, can still be used to change a disabled window's Z position.

Sometimes when using Accelerator Menu Commands to move a modal dialog window behind the disabled window that owns it, the modal dialog will immediately pop to the front, preventing its movement. This occurs when Automatic Activation is enabled and the mouse cursor is over the disabled window. When the modal dialog moves behind the disabled window, the disabled window tries to automatically activate. As with any disabled window, this causes its modal dialog to pop to the front and flash.

Z-Order Restrictions

Depending on the Z-Order Restrictions which are enabled, a Z movement Accelerator Menu Command may not move a window to the very back or the very front of the Z order. Instead, the window may only be moved as far back as its owner, and only as far front as its ownership siblings. Also, Z-Order Restrictions may cause the target window's owner and owned windows to move with it.

See Also

Accelerator Menu Controls

Accelerator Menu Commands

Accelerator Window Capture

Accelerator Menu Controls

These controls are part of the Windows Extensions (Group).

Accelerator menu on window border. (Checkbox)

This is an *option* that enables the Accelerator Menu Extension. When enabled, pressing the right mouse button while the mouse cursor is on a window border or title bar causes an Accelerator Menu to pop up. The Accelerator Menu contains Accelerator Menu Commands for window Z movement (Forward, Backward, etc.) and window show state (Minimize, Maximize, etc.).

Enable keyboard commands. (Checkbox)

This is a *sub-option* that enables the use of keyboard keys for some Accelerator Menu Commands. When enabled, pressing an Accelerator Menu Command key (Home, End, etc.) while the mouse cursor is on the window border or title bar causes the corresponding Accelerator Menu Command to be executed.

Capture window after operation. (Checkbox)

This is a *sub-option* that enables the capture of a window as the target of subsequent Accelerator Menu Commands (see Accelerator Window Capture). A window will be captured following any move, size or show state operation, or use of an Accelerator Menu Command key. A captured window is indicated by the cursor changing to a small white square on a black box. The window will remain captured until the mouse is moved or a mouse button is pressed.

Accelerator Menu Commands

The Accelerator Menu contains commands for controlling the Z position of a window, regardless of its show state (minimized, normalized, maximized). It also contains commands for changing a window's show state. Keyboard keys (shown below in parenthesis) can also be used for most Z position commands.

Depending on the current Z position and show state of the target window, not all commands may be valid. Invalid commands are shown as grayed in the Accelerator Menu. Invalid keyboard commands cause a beep.

Always On Top

This command toggles the window style of the target window between *normal* and *Always On Top*. An Always On Top window is placed in a Z order separate from that of the normal windows. The nature of this separate Z-order is such that its windows are always in front (on top) of the normal windows. As with normal windows, Always On Top windows can be moved backward and forward in their Z-order.

If a window is Always On Top, a check mark will appear to the left of this command in the popup Accelerator Menu.

Since Always On Top status can only be applied to a main window (e.g. the Program Manager window), this command is not available in the popup Accelerator Menu for an MDI Child window (e.g. a "group" window in the Program Manager window).

Front (Home)

Assuming no Z-Order Restrictions, this command moves the target window to the *front* of its Z order (i.e. normal or Always On Top). If after executing this command other windows are still in front, then the other windows are probably Always On Top windows (see Undo All "On Top" (Button)), or Z-Order Restrictions are enabled.

If the target is an MDI Child window, it will be moved to the front of its siblings' Z-order. Its main window (e.g. the Program Manager window itself) will not be affected.

Forward (Page Up)

This command moves the target window *forward* by one "significant" window regardless of the Z order it is in. If the only significant window in front of a normal one is Always On Top, then the target window will be moved in front of it and changed to Always On Top. A significant window is one which is visible and not minimized.

If the target is an MDI Child window, its main window will not be affected.

Backward (Page Down)

This command moves the target window *backward* by one "significant" window regardless of its Z order. If the only significant window in back of an Always On Top one is normal, then the target window will be moved in back of it and changed to normal. A significant window is one which is visible and not minimized.

If the target is an MDI Child window, its main window will not be affected.

Back (End)

Assuming no Z-Order Restrictions, this command moves the target window to the *back* of the normal Z order. An Always On Top window will be changed to normal.

If the target is an MDI Child window, it will be moved to the back of its siblings' Z-order. Its main window will not be affected.

Minimize

This command changes the target window to the *minimized*, or iconized, show state. It is identical in function to the Minimize command in the system menu, and the minimize button (down arrow) to the right of a window's title bar.

Even though a window is in a minimized show state, it can still be moved forward and backward in Z position.

Normalize

This command changes the target window to the *normal* show state. It differs from the Restore command in the system menu, which restores the window to its previous--normal or maximized--show state.

Maximize

This command changes the target window to the *maximized* show state. It is identical in function to the maximize button (up arrow) to the right of a window's title bar.

Even though a window is in a maximized show state, it can still be moved forward and backward in Z position.

Close/Exit

This command *closes* the target window and, possibly, causes its application to *exit*. It is identical in function to the Close command in the system menu, and double-clicking the system menu box to the left of the window's title bar.

Either the Close or the Exit command will appear in the Accelerator Menu. If closing the target window also causes the application to exit, then the Exit command will appear to warn the user.

Accelerator Window Capture

When moving a window backwards in Z, or when moving or sizing a window while maintaining its Z position (see Maintain Z Extension), it is possible for the target window to become lost behind other windows. To gain access to the window again, overlying windows must be moved out of the way or pushed back. This is especially a problem when using the Accelerator Menu keyboard commands to quickly adjust a window's Z position.

What Is It

If the Accelerator Window Capture sub-option is enabled in the Accelerator Menu Controls, a window will be *captured* after any size, move, or show state (minimize, normalize, maximize) change occurs to it. It will also be captured following any Accelerator Menu keyboard command.

Once captured, Accelerator menu and keyboard commands (see Accelerator Menu Commands) are directed to the captured window instead of the window under the mouse cursor. Also, pressing the mouse right button causes the Accelerator Menu to pop up whether or not the mouse is on a window border.

Capture Release

The target window will remain captured until the mouse is moved or a mouse button is pressed. Of course, pressing a mouse button sometimes can cause a new window operation, such as the appearance of a dialog box. If this happens then the original window will be released from capture and the newly shown window will be captured as the new Accelerator Window Capture target.

Capture Cursor

To indicate that a window has been captured, the cursor changes to a small white square on a black background. When capture is released the cursor will revert to its normal shape, typically an arrow pointer.

Maintain Z Extension

The Maintain Z Extension prevents the Z position (depth) of a window from changing when it is activated. This allows a window to be sized, moved, minimized, maximized, etc. without it first popping to the front on the desktop. Similarly, a window can be used (menus accessed, controls activated, contents scrolled) without it popping to the front.

This simplifies the arrangement of windows on the desktop and, more importantly, the maintenance of that arrangement. Besides reducing the amount of time wasted shuffling windows around each time one pops to the front, this allows windows to be used more effectively side-by-side since one window can be used even if partially cover by another.

Associated with the Maintain Z Extension is the ability to automatically activate main and MDI Child windows (see Automatic Activation). As the mouse moves over a window it is automatically activated without first clicking on it with the mouse. Eliminating the activation click helps to prevent spurious mouse actions and makes for a "snappier" user interface.

See Also

Maintain Z Controls

Automatic Activation

Maintain Z Controls

These controls are part of the Windows Extensions (Group).

Maintain Z position of activated window. (Checkbox)

This is an *option* that enables the Maintain Z Extension. When enabled, an activated window is prevented from popping to the front. Windows which pop up, such as a dialog box, are of course still allowed to pop to the front so that they may be easily seen and used.

Click border to pop to front (Shift to stop). (Checkbox)

This is a *sub-option* that enables the use of a left button mouse click on a window border or title bar to make the window pop to the front. The same action can be achieved with the Front Accelerator Menu Command. Clicking on a window that is already at the front causes a beep. Holding down the Shift key disables the pop-to-front action, which is convenient for maintaining Z after double-clicks or manual window activation.

When this sub-option is enabled, double-clicking on an icon (minimized window) will cause the window to first pop to the front (first click) and then open to its previous show state (second click). Similarly, double-clicking on a window title bar causes the window to pop to the front before its show state is changed.

Clicking on the border or title bar of an MDI Child window only pops it to the front of its siblings' Z order. To pop an MDI Child window to the front of all other windows, its main window must be brought to the front.

Auto activate main window under cursor. (Checkbox)

This is a *sub-option* that enables the Automatic Activation of the main window which is currently under the mouse cursor. This sub-option can be used without MDI Child window auto activation, in which case the main window is auto activated, but MDI Child windows must still be manually activated (clicked on to activate).

Auto activate MDI child windows. (Checkbox)

This is a *sub-option* that enables the Automatic Activation of the MDI Child window which is currently under the mouse cursor. This sub-option can be used without main window auto activation, in which case MDI Child windows will only be automatically activated if their MDI main window is active.

When using the menus on an MDI main window, the MDI Child window which is in front of its siblings, not the last one which was active, will remain active and serve as the target of the menu command (see Automatic Activation).

Automatic Activation

Normally, a window is activated by clicking anywhere on it. When a window activates it also pops to the front. Popping the activated window to the front is prevented by the Maintain Z Extension. Clicking on the window to activate it is called *manual* activation.

What Is It

Automatic activation permits a window to be activated by simply moving the mouse cursor over it. This helps to prevent inadvertent activation of window controls as a result of the activation click. It also makes for a "snappier" user interface since the window under the cursor is always ready for action--there is no sudden change in Z position and appearance when first trying to use the window.

Independent Sub-Options

The Maintain Z Extension provides sub-options for independently enabling and disabling main window and MDI Child window automatic activation (see Maintain Z Controls). If automatic activation for both main and MDI Child windows is enabled, then passing the mouse cursor over any window on the desktop will activate it.

If only main window automatic activation is enabled, then normal and MDI main windows will be automatically activated, but child windows of MDI main windows must be manually activated. If only MDI Child window automatic activation is enabled, then the MDI main window must be manually activated before its MDI Child windows will be automatically activated.

MDI Child Activation

Normally, when an MDI main window (e.g. the Program Manager window) activates, the MDI Child window (e.g. a "group" window in the Program Manager window) which was last active is also activated. Similarly, when using an MDI main window's menus, the last activated MDI Child window stays active and serves as the target of the menu command. This approach can **not** be used for automatic activation of MDI Child windows.

When automatic activation of MDI Child windows is enabled, the MDI Child window which is in front of its siblings will remain active when using its main window's menus. This approach, unlike the normal one, works even if other MDI Child windows lie between the target MDI Child window and the MDI main window menu. Otherwise, the last window which the mouse passed over on the way to the menu would remain active, which could be the wrong target window.

Default Activation

Windows insists that some main window on the desktop always be activated. If main window automatic activation is enabled and the mouse cursor is on the desktop--not a window--then the WinX main window will be activated as the default. Similarly, if the mouse cursor is on a disabled window (see below) then the WinX main window is also used as the default activation window.

Disabled Windows

A disabled window occurs when a modal dialog that it owns pops up. For example, if the New item is selected in the Program Manager's File menu, then a New Program Object dialog window will pop up. This dialog is a modal dialog, which means that as long as it is open the window which owns it--the Program Manager--can not be activated or used.

As mentioned above, Windows requires that some window always be activated. Since a disabled window can not be activated the WinX main window is used as the default activation window. Thus, whenever the mouse cursor passes over a disabled window, the window's modal dialog will pop to the front and flash, and the WinX main window will activate. Trying to click on a disabled window causes the same action, as well as a beep for emphasis.

Size/Move Border Extension

The Size/Move Border Extension allows a window to be moved, as well as sized, by dragging its border. Dragging a window side border, or the title bar, moves the window. Dragging a corner of the window border sizes the window. The mouse cursor shape changes to provide feedback to the user as to the expected operation.

Moving a window with its border permits the window to be moved even though its title bar is covered. This is especially important since WinX allows windows to be moved and sized without first popping them to the front (see Maintain Z Extension). Use of the border also permits windows to be moved off the top edge of the display, instead of just the bottom and side edges.

Mouse Cursor

When the mouse cursor is on a side border (top, bottom, left, or right) the cursor shape changes to a four-arrow "move" cursor. When the mouse cursor is on a border corner (top-left, top-right, bottom-left, bottom-right) its shape changes to a two-arrow "size" cursor. When on the title bar, although the expected operation is to move the window, the mouse cursor will retain its normal shape, the arrow pointer.

Shift Key

Holding down the Shift key before starting a size or move operation swaps the type of operation which will be performed. Dragging a side border will size the window, and dragging a border corner will move it. The cursor shape will change accordingly to reflect the operation which is to be expected.

See Also

Size/Move Border Controls.

Size/Move Border Controls

These controls are part of the Windows Extensions (Group).

Size/move window border (Shift to swap). (Checkbox)

This is an *option* that enables the Size/Move Border Extension. When enabled, a window can be moved by dragging a side border as well as its title bar, and it can be sized by dragging a border corner. Holding down the Shift key swaps the type of operation performed on a border area. The shape of the cursor on the window border always reflects the operation (size or move) which will occur.

Close Menus Extension

The Close Menus Extension forces dropdown and popup menus to always close when the initiating mouse button is released. Normally, clicking on a dropdown menu causes the menu to open and stay up. The menu will only close if a valid item is selected or something else on the desktop is clicked. A similar situation often occurs with popup menus.

Keeping menus up after an initiating click is a useful feature for novices. As one gains greater facility with using the mouse, however, this can become an aggravation. The additional action of sometimes having to click a second time to close a menu can make menus seem cumbersome. This is one of the reasons why keyboard commands often seem quicker.

Closing a menu when the initiating button (i.e. left button for a dropdown menu, right button for a popup menu such as the WinX Accelerator Menu) is released gives the user interface a "snappier" feel. Menus seem less cumbersome to use. It also eliminates possible spurious actions caused by clicking the mouse a second time to close the menu.

Exceptions

Because of their unique nature, system menus on icons (minimized main windows) and dropdown lists on combo boxes will not be affected by the Close Menus extension.

If snappy interaction with icons is desired then use the right button to pop up an Accelerator Menu (see Accelerator Menu Extension), which contains commands similar to those found in the system menu.

See Also

Close Menus Controls.

Close Menus Controls

These controls are part of the Windows Extensions (Group).

Close menus on mouse button release. (Checkbox)

This is an *option* that enables the Close Menus Extension. When enabled, dropdown and popup menus will be closed when the initiating mouse button is released. Because of their special nature, this action does not apply to system menus on icons, or dropdown lists on combo boxes.

Z-Order Restrictions (Group)

The Z-Order Restrictions Group in the WinX main window provides a set of *options* (checkboxes) and *sub-options* (indented checkboxes) which enable and disable restrictions to the handling of window Z (depth) movement. A restriction is somewhat like an extension, only with a negative effect. Disabling an option also disables the functionality of its sub-options. Its sub-options, however, can still be checked and unchecked.

In Windows, a window can "own" other windows. A simple example of this is a main window owning its popup dialog windows. Since Windows does not allow the user to control window Z position, ownership is generally not of concern to the user. WinX allows windows to be moved forward and backward in Z (see Accelerator Menu Extension). This brings up the question of what to do with the owner and owned windows of the window being moved (see Owner Behind Restriction)

Restriction Usage

We suggest that WInX be used with no restrictions. This allows any window (except MDI Child windows) to be moved anywhere in Z, with no other windows being affected. This is generally not a problem since ownership relations are usually minimal in Windows applications, and since WinX allows the user to easily push back windows to expose lost ones.

A lack of restrictions, however, can be disconcerting to the novice, and sometimes inconvenient to the expert. Z order restrictions can be used to keep windows together in Z the same way that Multiple Document Interfaces (MDI) are used to keep windows together in X and Y.

Grayed Options

Restrictions are only in effect when the options and sub-options in the Windows Extensions (Group) which they affect are enabled. When a restriction is not in effect its corresponding option (checkbox) will be disabled and its text appear grayed.

See Also

Refer to the following topics and sub-topics for a description of each Z-Order Restriction, its effect on Windows, and its option and sub-option controls.

Owner Behind Restriction

- Owner Behind Controls

Owner Behind Restriction

The Owner Behind Restriction forces a window to stay behind any windows it owns. For example, a main window could never be moved in front of one of its dialog boxes. Similarly, a dialog box could never be moved behind its main window. The Accelerator Menu Commands that would cause such an action would appear grayed.

A further restriction, available as a sub-option, forces all owned windows to stay together (see Owner Behind Controls). For example, another window unrelated by ownership could never be moved in-between a main window and its dialog box.

Z Order Normalization

Whenever the Owner Behind Restriction is enabled or, if already enabled, any of its sub-options are enabled, the Z order of all the windows on the desktop will be *normalized*. Normalization of the Z order entails moving windows (recursively) in Z such that owner windows are all behind their owned windows, and owned windows are all together. This assures a known good starting point before handing over Z control to the user.

During normalization, windows which are not owned by another window will maintain their relative Z position on the desktop. These windows and any windows which they own will also retain their Always On Top status (see Accelerator Menu Commands).

See Also

Owner Behind Controls.

Owner Behind Controls

These controls are part of the Z-Order Restrictions (Group).

Owner always behind owned windows. (Checkbox)

This is an *option* that enables the Owner Behind Restriction. When enabled, if a window is moved forward, any windows that it owns directly or indirectly also are moved forward. For example, moving the Program Manager window forward will also move any dialog box it may have open.

If a window is moved backward, and no sub-options are enabled, no other windows are affected since it will still remain behind any of its owned windows. The window, however, can only be moved back as far as its owner. Thus, if a window has an owner, selecting the Back Accelerator Menu Command will move it back to just in front of its owner.

This option will be disabled and appear grayed if the **Accelerator menu on window border** option in the Accelerator Menu Controls, or the **Click border to pop to front (Shift to stop)** sub-option in the Maintain Z Controls is enabled.

Owned windows kept together. (Checkbox)

This is a *sub-option* that keeps owned windows together, thereby preventing unrelated windows from coming in-between a window and its owner. When enabled, a window will only be allowed to move to the front of its ownership siblings--just in front of any other windows which are owned directly or indirectly by the same owner.

If an unrelated window is moved forward or backward, it will skip over clusters of windows which are related by ownership. For example, the File Manager window will skip over the Program Manager Window and any dialog box it may have open.

Include MDI children. (Checkbox)

This is a *sub-option* that enables the inclusion of MDI Child windows in any ownership restrictions. MDI Child window restrictions will be limited to the Z order of their siblings, not that of the main window. Usually MDI Children do not own other windows. The mechanism, however, is there in Windows and some applications may choose to take advantage of it.

Save Settings... (Button)

The Save Settings button in the WinX main window causes the Save Settings (Dialog) window to pop up. This dialog provides options for saving the WinX control settings and window placement.

See Also

Save Settings (Dialog)

Save Settings (Dialog)

The Save Settings dialog provides options for saving the WinX control settings and window placement. Settings can be saved immediately, or later upon application exit.

WinX application settings consist of the state of all options and sub-options in the Windows Extensions Group and the Z-Order Restrictions Group, the state of the save options in this dialog, and the position and show state of the main window.

Settings are saved in the application initialization file *winx.ini*, which is located in the Windows directory, typically *c:\windows*. Whenever WinX is started it is automatically initialized from the information in this file. There is generally no need to edit this file directly. It should only be updated through WinX.

Save Settings On Exit (Checkbox)

This *option* enables the saving of settings whenever the WinX application exits. Exit can be initiated directly by the user (the Close command in the system menu, or the Exit Accelerator Menu Command), or indirectly by quitting Windows. Settings are generally not saved if Windows hangs or crashes (one of the reasons for the **Save Now** button).

Done (Button)

This button causes the dialog to close, and any changes to its controls to take effect. It does **not** cause the settings to be saved.

Save Now (Button)

This button causes the dialog to close, any changes to its controls to take effect, and the application settings to be saved immediately. This button allows settings to be conveniently saved without exiting WinX or waiting for Windows to terminate.

This is the dialog's *default* button. Hitting the Return key is equivalent to pressing this button.

Cancel (Button)

This button causes the dialog to close, and cancels any changes to its controls.

Hitting the Escape key is equivalent to pressing this button.

Standard Settings... (Button)

The Standard Settings button in the WinX main window causes the Standard Settings (Dialog) window to pop up. This dialog provides selections which configure the options and sub-options in the Windows Extensions (Group) and Z-Order Restrictions (Group) to a known state, from pre-defined or saved settings.

See Also

Standard Settings (Dialog)

Standard Settings (Dialog)

The Standard Settings dialog provides selections which configure the options and sub-options in the Windows Extensions (Group) and Z-Order Restrictions (Group) to a known state, from pre-defined or saved settings. The save options (see Save Settings (Dialog)) and the window placement are not affected by the selections.

The selection of standard settings is provided by a set of radio buttons (i.e. making a selection undoes any other selection). The selected settings are shown immediately by the option and sub-option checkboxes in the WinX main window. They do not take effect, however, until the dialog is closed.

Disable WinX (no Options or Sub-options) (Radio Button)

This *selection* disables all the options and sub-options in the Windows Extensions (Group) and the Z-Order Restrictions (Group). This completely disables WinX without having to exit the application. It also provides a convenient starting point for gradually enabling options and sub-options.

Full Capability (all Extensions, no Restrictions) (Radio Button)

This *selection* enables all options and sub-options in the Windows Extensions (Group), and disables the options in the Z-Order Restrictions (Group). This completely enables the full capabilities of WinX. This is also the default configuration used when WinX is first installed.

Retrieve Saved Settings (.INI File) (Radio Button)

This *selection* retrieves the control settings for the options and sub-options in the Windows Extensions (Group) and the Z-Order Restrictions (Group) which are currently saved in the WinX initialization file (see Save Settings (Dialog)).

Done (Button)

This button causes the dialog to close, and any changes to the application's controls to take effect. It does **not** cause the settings to be saved.

This is the dialog's *default* button. Hitting the Return key is equivalent to pressing this button.

Save Now (Button)

This button causes the dialog to close, any changes to the application's controls to take effect, and the application settings to be saved immediately. This button allows settings to be conveniently saved without exiting WinX or waiting for Windows to terminate.

Cancel (Button)

This button causes the dialog to close, and cancels any changes to the application's controls.

Hitting the Escape key is equivalent to pressing this button.

Undo All "On Top" (Button)

The Undo All "On Top" button in the WinX main window changes the status of any *Always On Top* windows on the desktop to *normal* (see Accelerator Menu Commands).

This is the main window's *default* button. Hitting the Return key is equivalent to pressing this button.

About... (Button)

The About button in the WinX main window causes the About (Dialog) window to pop up. This dialog provides general information about the WinX application.

See Also

About (Dialog)

About (Dialog)

The About dialog provides general information about the WinX application. This is the same text provided in About WinX. Its contents includes:

- Product name, version and copyright notice for WinX.
- A brief description of the purpose of WinX.
- Shareware registration, author's name and address.

OK (Button)

This button causes the dialog to close.

This is the dialog's *default* button. Hitting the Return key is equivalent to pressing this button. Hitting the Escape key is also equivalent to pressing this button.

Help... (Button)

the Help button in the WinX main window causes the Windows Help window to pop up with the WinX help manual. Basic instructions for using the Windows Help system are provided at the beginning of WinX Help Contents.

For a more thorough description of the Windows Help system, consult your Windows documentation.

No Search

Please note that the *search* facility of the Windows Help system is not enabled for WinX (i.e. Search button grayed). A conscious decision was made not to provide keywords for its use.

It is the opinion of the author that the keyword search facility provided by the Windows Help system is of extremely limited use, and can actually be counterproductive. With keyword searches an author invariably provides all the necessary keywords, except those needed by the user. This results in frustration and can turn off users to the idea of using online help.

A future product of Peculiar Technologies will address this problem.

Solving Problems

Windows wants the user to interact with menus, controls, and especially windows in a very specific manner. The user is provided with few options. This philosophy is reflected in the software underpinnings of Windows.

Although seemingly simple and unobtrusive, the features provided by WinX have had to rely on the software equivalent of a 2x4 to convince Windows to do things a bit differently. Using this approach in software, as with mules, has its drawbacks.

The Nature of Problems

Every effort has been taken in the design of WinX towards the goal of peaceful coexistence with all other applications. In application development there are many paths to essentially the same destination. Most developers choose a well-traveled path, one that the Windows developers perhaps intended. For often very good reasons a different path sometimes must be chosen.

It is virtually impossible to anticipate every path that developers may have chosen in building their applications. WinX anticipates many of these, and does an excellent job of working with most applications. Occasionally, however, there are problems.

Some applications insist on being activated only with the mouse, others manhandle the cursor shape, and still others kidnap the mouse or keyboard for their exclusive use. These and other problems can cause conflicts with WinX, which is trying to do a little of this itself.

The Nature of WinX

A mechanism called *hooks* was provided by the Windows developers to give limited access to the inner workings of Windows. In WinX, each of the Windows Extensions uses its own set of system-wide hooks to monitor system message traffic. As needed, the extension's hooks intercept and translate these messages to effect a particular change in user interface behavior.

Each extension can be enabled and disabled by means of an option (checkbox) in the Windows Extensions (Group) in the WinX main window. Disabling an option disables the extension and its corresponding hooks. The hooks are removed from Windows and their monitoring and modification of system message traffic ceases. The other extensions and their hooks, however, are left unaffected.

An extension also may have sub-options (indented checkboxes). Sub-options are implemented by flags in the extension's hook software. Disabling a sub-option causes sections of the hooks' code to stop executing. Because of possibly unforeseen interactions between the message traffic and the hook code, disabling a sub-option can be less definitive than disabling an option.

The Nature of Solutions

If a conflict between WinX and a particular application is suspected, first try disabling sub-options which are related to the problem. If that fails, then try disabling whole options to isolate the problem. If all else fails, select **Disable WinX** in the Standard Settings (Dialog).

If disabling WinX fails to clear up the problem then WinX is not at fault. The author has experienced, even with mainstream applications, occasional intermittent problems such as windows not scrolling, cursor shapes and menus getting stuck, button clicks being missed, and Windows itself freezing--and WinX was not even running.

Problems Which Affect WinX

Some problems with WinX only affect using features of WinX. Common ones include:

- No Automatic Activation of pseudo-MDI Child windows.
- No Accelerator Window Capture cursor.
- Accelerator Window Capture of the wrong window.
- No response to Accelerator Menu Extension command keys.
- No response to the Size/Move Border Extension Shift key.

These problems can either be ignored, or the corresponding option or sub-option can be disabled to

avoid an inconsistent user interface.

Problems Which Affect Other Applications

Some problems with WinX affect using other applications. Common ones include:

- Localized failure of Accelerator Window Capture to release.
- Newly opened windows closing prematurely, or failing to close normally.
- Newly opened pseudo-MDI Child windows failing to pop to the front.

Capture problems can be solved by disabling the capture sub-option in the Accelerator Menu Controls. Many of the problems concerning newly opened main windows can be solved by disabling the Automatic Activation sub-option for main windows in the Maintain Z Controls. See Known Problems for issues concerning child windows in pseudo-MDI applications, such as Microsoft Word and Excel.

See Also

Known Problems

Known Problems

Due to the impossibility or extreme difficulty of fixing them, certain problems in using WinX are known to exist but have not been fixed. Some the user can probably live with quite comfortably. Others the user must carefully avoid or, trading WinX capability for safety, certain options and sub-options in WinX must be disabled.

MS-DOS Prompt

When an MS-DOS Prompt window is active, all features of WinX which rely on keyboard input will be temporarily lost. Lost features include key commands for the Accelerator Menu Commands, and use of the Shift key in the Size/Move Border Extension. The MS-DOS Prompt window which Windows provides for running DOS applications is unique. Unlike a normal application, it eats all keyboard input before even a keyboard hook can get to it.

Another problem involves using Alt+Enter to toggle between Windows and full-screen modes. If the automatic activation sub-option for main windows in the Maintain Z Controls is enabled, when MS-DOS Prompt tries to toggles to full-screen mode its window will minimize and it will remain in Windows mode. If the newly minimized window is double-clicked, or the Maximize Accelerator Menu Command is selected, then MS-DOS Prompt will switch to full-screen mode.

Finally, if the MS-DOS Prompt window is active, the screen saver will not activate. This is a feature of MS-DOS Prompt, not WinX.

Visual C++

On the toolbar of the Visual C++ application main window there is a Project Files Button (the far left button icon). Clicking this button causes a list of project files to drop down. The list is a separate window, not a combox dropdown list. As with any other window in WinX, if the list window is open and the main window is moved forward, it will be covered up by the main window. Pushing the main window backwards will expose the list window, or the owner behind option in the Owner Behind Controls can be enabled to prevent the list from going behind its main window owner.

Also, if the automatic activation sub-option for main windows in the Maintain Z Controls is enabled, the only way to close the list is by moving the mouse cursor into and then out of the window--a second click on the icon button will not work.

Pseudo-MDI Applications

WinX is unable to perform Automatic Activation of child windows belonging to *pseudo* MDI applications, such as Microsoft Word and Excel. WinX is also unable to assure that their child windows pop to the front upon opening. Although some applications appear to be Multiple Document Interface (MDI) applications, internally they are not. Another application, such as WinX, is therefore unable to effect or detect the activation of their child windows since the activation process is private.

Accelerator Menu

There is an apparent bug in the manner used by Windows to position popup menus under certain conditions. Thus, whenever the user pops up an Accelerator Menu (see Accelerator Menu Extension) near the left hand side of the display, the menu will appear under the mouse cursor instead of above, below, or to the right of it. This often causes a menu item to be inadvertanly selected and executed. Being unexpected, this action may appear to the user as some random bug in WinX.

See Also

Questions and Comments

Questions and Comments

Questions and comments are welcome. See About WinX for the mail and e-mail addresses where the author can be contacted.