WinX Help Contents

Welcome to the Help system for the *WinX: Windows User Interface Extensions* utility. 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 underlined words--Getting Started--to get started.

Overview

- Getting Started
- Why Use It
- About WinX
- Disclaimer and Trademarks

Main Window Controls

- Windows Extensions (Group)
- Z-Order Restrictions (Group)
- Undo All "On Top" (Button)
- Settings... (Button)
- Hide WinX (Button)
- Enable/Disable WinX (Button)
- Help... (Button)
- About... (Button)

In Case of Trouble

- Solving Problems
- Known Problems
- Questions and Comments

Getting Started

Since most Windows applications perform common windowing functions in similar ways, it is often possible to use a new application without reading its help manual. The whole point of WinX, however, is to allow the user to perform windowing functions in new and better ways. That being the case, a quick review of this help manual prior to using WinX is highly recommended.

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 (press, release) the **left** mouse button pops the window to the front. Pressing (and holding down) the **right** mouse 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 or run hidden--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). Even scrollable windows which are partially covered can be used (use keyboard Page Up, Page Down and the arrow keys to scroll). Maintaining window Z position also allows an individual window to 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 Windows 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, Automatic Activation activates and deactivated windows as the mouse passes over them. 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 technique is often called "sticky menus". Although a useful feature for the novice, it can become annoying as one gains greater facility with the mouse. It also 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).

WinX can be run hidden, and more:

The Hide WinX Button allows WinX to be run "hidden"--with no icon or main window appearing on the desktop. The Undo All "On Top" Button changes the status of all "Always On Top" windows on the desktop to normal. The Enable/Disable WinX" Button temporarily disables all of WinX, and then allows it to be re-enabled. WinX settings can be set to a known state or saved with the Settings Dialog. This dialog also allows WinX to be run hidden at startup, and settings to be saved on exit.

WinX is adapted to work with Windows:

There are inherent differences between the WinX and Windows user interface paradigms. Features have been included in WinX to help minimize these differences. The Accelerator Capture feature allows a window which becomes covered as a result of a move, size, or Z movement operation to be retrieved. It also works with Automatic Activation to allow accelerator and shortcut keys to be used on a popup window even though the cursor is not on the window.

Some windows are "shy". Automatic Activation can cause them to disappear when they deactivate. Holding down the Shift key prevents activation from changing until the mouse can be moved to them.

In Windows, the last active MDI child stays active when using its main window menus. Automatic Activation conflicts with this. Instead, WinX keeps the front MDI child active.

About WinX

Name, Version, Copyright

WinX: Windows User Interface Extensions
Version: 1.5
Copyright© 1993-1994, Peculiar Technologies

Description

WinX extends Windows 3.1 to provide workstation-like windowing features. Any window can be moved in Z, dragged by its border, used without popping to the front, and made Always On Top. Also makes all menus "unsticky". (See also Getting Started and Why Use It.)

Distribution, Questions, Comments

WinX is distributed as shareware. You may evaluate it for up to 30 days after which time it must be either registered or removed from your system. If you find WinX useful please register it by sending the product name, version, and $20 to its author:

Jon Barrilleaux
3800 Lake Shore Ave.
Oakland, CA 94610

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

(Internet)	71461.432@compuserve.com
(Compuserve)	71461,432

Disclaimer and Trademarks

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
  - Accelerator Menu Controls
  - Accelerator Menu
  - WinX Menu
  - Accelerator Capture
- Maintain Z Extension
  - Maintain Z Controls
  - Automatic Activation
- Size/Move Border Extension
  - Size/Move Border Control
- Close Menus Extension
  - Close Menus Control

Accelerator Menu Extension

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

Commands are accessed in the popup Accelerator Menu by holding down 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. the Accelerator Menu and key commands can also be accessed if Accelerator Capture is active regardless of where the cursor is.

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 to the left of the window title bar).

WinX Menu

Also included in the Accelerator Menu is a cascade (pull right) WinX Menu for easy access to the WinX main window and its functions.

Accelerator Capture

Associated with the Accelerator Menu is the concept of Accelerator Capture . Accelerator Capture is a feature which allows Accelerator Menu commands to be used on a window to retrieve it if it inadvertently becomes hidden as a result of being moved or sized, or being pushed backwards. Accelerator Capture also allows accelerator and shortcut keys to be used on a window by temporarily keeping a newly activated window active even if automatic activation (see Maintain Z Extension) is enabled and the mouse is not on the window.

MDI Child Windows

Main windows, such as the Program Manager window, and Multiple Document Interface (MDI) Child windows, such as the Group windows in the Program Manager window, 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 and any of its Group windows--its MDI Child windows.

Z-Order Restrictions

Depending on the Z-Order Restrictions which are enabled, an Accelerator Menu Z movement command may not move a window to the very back or the very front of the Z order. Instead, the window may be moved only as far back as its owner, and as far front as its ownership siblings (i.e. windows with the same owner, such as two non-modal dialog windows belonging to the same application). 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
WinX Menu
Accelerator 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 commands for window Z movement (Forward, Backward, etc.) and show state change (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 if changed (Shift to hold).   (Checkbox)

This is a *sub-option* that enables the capture of a window as the target of subsequent Accelerator Menu commands (see Accelerator Capture). A window will be captured when it activates, when it is moved or sized, or when it is moved in Z with an Accelerator Menu command key. In general, capture will not occur if the cursor is already on the target window's title or border. While capture is in affect: the cursor will change to a pointer with a small white square on a black box; activation will be prevented from changing; and, popup and keyboard Accelerator Menu commands will be directed to the captured window regardless of the mouse cursor position.

A window will remain captured until the mouse is moved, a mouse button is pressed, or the Shift key alone is pressed and released. Holding down the Shift key, by itself, prevents capture from being lost regardless of mouse or keyboard activity.

Accelerator Menu

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 and for closing a window. Keyboard keys (shown below in brackets) 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 menu items in the Accelerator Menu. Invalid keyboard commands cause a beep.

WinX

Selecting this menu item causes the WinX Menu to pop up.

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. WinX, unlike many other applications, tests the actual status of the target window in determining whether or not to display the check mark.

Since Always On Top status can only be applied to a main window (e.g. the Program Manager window), this command does not appear 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 (normal or Always On Top). If after executing this command other windows are still in front, then the other windows may be Always On Top windows (see Undo All "On Top" (Button)), the target window may be an MDI child window, or Z-Order Restrictions may be enabled.

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

Forward [Page Up]

This command moves the target window *forward* by one "significant" window regardless of the Z order (normal or Always On Top) it belongs to. 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 the Z order (normal or Always On Top) it belongs to. 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* (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 show state was--normal or maximized.

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. Note that if an MDI child window is maximized, Windows will maximize and pop to the front any sibling which becomes active. Main windows do not exhibit this unusual behavior.

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 as a warning to the user.

WinX Menu

The WinX Menu is a cascade (pull right) menu in the Accelerator Menu. It provides easy access to the WinX main window and its functions even if the main window is hidden (see Hide WinX Button), minimized, or covered.

Undo All On Top

Performs the same action as the Undo All "On Top" Button.

Show/Hide WinX

If the WinX main window is hidden (i.e. invisible, see Hide WinX Button) then **Show WinX** will appear. This command will make the WinX main window visible, normalize it and pop it to the front. If the WinX main window is not hidden then **Hide WinX** will appear. This command will make the WinX main window invisible.

Disable WinX

Performs the same action as the Disable WinX Button, with the addition that the WinX main window will normalize and pop to the front. Since this menu can not be accessed when WinX is disabled, popping up the WinX main window prior to to WinX being disabled conveniently allows WinX to be re-enabled.

Settings...

Performs the same action as the Settings... Button.

Help...

Performs the same action as the Help... Button.

About...

Performs the same action as the About... Button.

Exit WinX

Causes the WinX main window to close and the WinX utility to exit immediately.

Accelerator 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.

When automatic activation is enabled (see Maintain Z Extension) and a new window pops up, a conflict occurs. Should the new window be activated so its accelerator and shortcut keys can be used without having to first move the mouse cursor onto the window, or should the window under the mouse remain activated.

What Is It

Accelerator Capture is a feature which allows Accelerator Menu commands to be used on a window to retrieve it if it inadvertently becomes hidden as a result of being moved or sized, or being pushed backwards. Accelerator Capture also allows accelerator and shortcut keys to be used on a window by temporarily keeping a newly activated window active even if automatic activation (see Maintain Z Extension) is enabled and the mouse cursor is not on the window.

When a window is captured: accelerator menu and keyboard commands (see Accelerator Menu) are directed to the captured window instead of to the window under the mouse cursor; pressing the mouse right button causes the Accelerator Menu to pop up whether or not the mouse is on the captured window's title or border; and, Automatic Activation is temporarily suspended. While capture is active, the mouse and keyboard can still be used normally, with the exception that controls associated with the Accelerator Menu (mouse right button, keyboard command keys) will not intercepted by WinX and not passed on to the active window.

Capture Initiation

If the Accelerator Capture sub-option is enabled in the Accelerator Menu Controls, any window which activates will also be captured. Capture also occurs, regardless of activation, if a window is moved or sized, or if an Accelerator Menu command key is used.

Since capture interferes with the normal cursor shape, and since it is not needed if the target window can still be accessed with the mouse cursor, capture will not occur if the cursor is still on the target window's title or border following the operation (activation, move, size, etc.).

If an MDI main window activates, the main window will be captured initially, not its active MDI child. Once the main window is active, any MDI child which activates will be captured.

Capture Release

The target window will remain captured until the mouse is appreciably moved (same tolerance as for mouse double-clicking) or a mouse button is pressed. Capture will also be released if the Shift key, by itself, is pressed and released.

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 Capture target.

Capture Hold

Holding down the Shift key before or during capture *holds* the capture, preventing it from being lost regardless of mouse or keyboard activity, or window activation. This is especially convenient when trying to popup the Accelerator Menu with the right mouse button, which can cause the cursor to move and capture to be lost, or when chaining Accelerator Menu commands.

For example, on an icon, press the Shift key and leave it pressed. Popup the Accelerator Menu by pressing the right mouse button and select Normalize. Without moving the mouse, press the right mouse button again and select Back from the Accelerator Menu. Since the target window is captured when it normalizes, and stays captured because the Shift key is pressed, the Accelerator Menu can be popped up without the mouse being on the target window's title or border.

Capture Cursor

To indicate that capture is in effect, the cursor changes to a arrow pointer with a small white square on a black background. The pointer helps emphasize the fact that during capture the mouse can still can be used normally. When capture is released the cursor reverts to the shape dictated by the underlying window, typically a simple 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. Note that if a scrollable window's scrollbar is covered then keyboard keys (Page Up, Page Down, arrow keys) often can be used to scroll the window.

Maintaining window Z position 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 covered by another.

Automatic Activation

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. Modal dialogs, such as for file open, are still allowed to pop to the front so that they may be easily seen and used.

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

This is a *sub-option* that enables the use of a left button mouse click (press and release) on a window border or title bar to make the window pop to the front. The same action can be achieved with the Front command in the Accelerator Menu. To inform the user that a window is as far front as it can go even though other windows are still in front of it, a beep will occur.

Holding down the Shift key *holds* the Z position by disabling the pop-to-front action. This is convenient for maintaining Z after double-clicks to change window show state, and after single clicks for manual window activation.

For windows which have no border or title bar, clicking anywhere on the window will cause it to pop to the front. Holding down the Shift key allows such windows to be used without them popping to the front.

Single-clicking on an icon (minimized window) causes the system menu to popup without the icon popping to the front. Double-clicking on an icon causes it to pop to the front before opening to its previous show state. 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.

Allow system to pop windows to front. (Checkbox)

This is a *sub-option* that allows the system to pop windows to the front. In general, WinX blocks most window Z position changes that are not directly initiated by the user via WinX. Enabling this sub-option allows the system to pop windows to the front as a result of system commands (Alt+Tab, Task List, Ctl+Esc, etc.), an application button push or menu selection, or screen saver startup.

Auto activate main window under cursor (Shift to hold). (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 to activate). Holding down the Shift key, by itself, *holds* the activation by preventing it from changing.

Auto activate MDI child windows (Shift to hold). (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 be automatically activated only if their MDI main window is active. Holding down the Shift key, by itself, *holds* the activation by preventing it from changing.

When using the menus or controls 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 is a feature which 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 usual 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 (e.g. the Program Manager window) will be automatically activated, but child windows of MDI main windows (e.g. a Group windows in the Program Manager window) 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 or controls, 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.

Unavoidably, the user must be especially careful when the MDI Child windows are tiled because it is not at all obvious which one is in front of its siblings. Regardless of the situation, however, the target of an MDI main window control action will be the MDI Child window which appears active. Using a noticeable color for the active window border, such as bright red, will help this situation.

Disabled Windows

A window is typically disabled when a modal dialog that it owns, directly or indirectly, pops up. For example, if New 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.

Windows requires that some window always be activated. Since a disabled window can not be activated, its popup window is activated instead as the default. To avoid losing modal dialog windows, whenever a modal dialog is used as the default it is first popped to the front and flashed. Thus, if auto-activation is enabled and the mouse cursor passes over a disabled window, its modal dialog will pop to the front, flash, and activate. Clicking on a disabled window causes the same action, as well as a beep for emphasis.

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 then its enabled popup window will be activated as the default. If a disabled window has no enabled popup window, then the WinX main window will serve as the default activation window.

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, by itself, 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 Control.

Size/Move Border Control

This control is 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, by itself, *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 drop down and popup menus to always close when the initiating mouse button is released. Normally, clicking on a drop down 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 drop down 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 windows) and drop down 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 the Accelerator Menu, which contains commands similar to those found in the system menu (Minimize, Maximize, etc.).

See Also

Close Menus Control.

Close Menus Control

This control is 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, drop down 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 drop down 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, however, 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

In general we suggest that WinX be used with the **Owner always behind owned windows** option in the Owner Behind Controls enabled (i.e. the default upon WinX installation). This is a compromise between no restrictions, which can lead to problems with some applications, and full restrictions, which can be overly restrictive.

With no restrictions, any window (except MDI Child windows) can be moved anywhere in Z, with no other windows being affected. This is generally not a problem since ownership relations are 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.

Of greater concern, however, is a problem which occurs with some applications where a dialog or information window gets stuck behind its owner (see **Owned Windows** in Known Problems). The **Owner always behind owned windows** option prevents this.

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, and the Accelerator Menu command 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).

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 be behind 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 command in the Accelerator Menu 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 and the **Click border to pop to front (Shift to hold)** sub-option in the Maintain Z Controls are disabled.

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.

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).

Settings...   (Button)

The Settings button in the WinX main window causes the Settings (Dialog) window to pop up. This dialog provides standard selections which configure the WinX control settings to a known state, from pre-defined or saved settings. The dialog also provides options for running WinX hidden at startup, and for saving settings upon exit.

See Also

Settings (Dialog)

Settings (Dialog)

The Settings dialog provides standard 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 dialog's save options and the window placement are not affected by these selections. The dialog also provides options for running WinX hidden at startup, and for saving settings upon 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.

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 in the WinX main window by its option and sub-option checkboxes. They do not take effect, however, until the dialog is closed.

Clear (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). It also completely disables WinX without having to exit the application (the same as for the Disable WinX Button), and provides a convenient starting point for gradually enabling options and sub-options.

Default (same as upon installation) (Radio Button)

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

Saved (retrieved from .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.

Run Hidden On Startup (Checkbox)

This *option* causes the WinX application to run hidden upon startup. When running hidden neither the main window or its icon will appear on the desktop, and WinX will not appear in the Windows Task List. WinX can be made visible by running it again, such as from the Program Manager, or by selecting **Show WinX** in the WinX Menu.

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 (Close command in the system menu, Exit command in the Accelerator Menu), 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.

Hide WinX (Button)

The Hide WinX button in the WinX main window immediately hides WinX, which then continues to run hidden. This is equivalent to selecting **Hide WinX** in the WinX Menu. WinX can also be configured to run hidden automatically at startup (see Settings Dialog).

When running hidden, neither the main window or its icon will appear on the desktop, and WinX will not appear in the Windows Task List. WinX can be made visible by running it again, such as from the Program Manager, or by selecting **Show WinX** in the WinX Menu.

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

Enable/Disable WinX (Button)

The Enable/Disable WinX button in the WinX main window toggles WinX between an enabled and a disabled state. When WinX is enabled, the button will read **Disable WinX**. When WinX is disabled, the button will read **Enable WinX**.

When WinX is disabled: all options and sub-options will be disabled and grayed in the WinX main window; all buttons will be disabled except Enable WinX, Help, and About; and, all extensions will be removed from Windows the same as if WinX had been exited. When WinX is re-enabled, the previous settings will be restored and all controls will be un-grayed.

WinX exit and start up action will be unaffected if WinX is exited while disabled.

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.

The Windows Help window for WinX will also pop up if the WinX window is active and the F1 key is pressed.

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

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 WinX.
- Shareware description, registration, author's name and address.

Registration Number (Field)

If your copy of WinX is unregistered, the About dialog will popup whenever WinX starts up or exits. This serves as a reminder that WinX is distributed as shareware and that it is a violation of the law to use it for longer than the trial period without registering it.

When you register your copy of WinX, you will receive a registration number to type into this field. Once the correct registration number is entered, the About dialog will no longer popup automatically as a reminder.

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.

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 and certain assumptions made by some Windows applications.

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 developers of Windows 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 kidnap the mouse or keyboard for their exclusive use, and still others use "transparent" and "magic" windows to achieve special effects. These and other problems can cause conflicts with WinX.

The Nature of WinX

A mechanism called *hooks* was provided by the developers of Windows 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, use the Disable WinX Button in the WinX main window, or select the Disable WinX item in the WinX Menu

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

- No Automatic Activation of pseudo-MDI Child windows.
- Accelerator Capture cursor disappears.
- No response to Accelerator Menu command keys or the Shift key.

These problems can either be ignored, or the corresponding option or sub-option can be disabled to avoid an inconsistent user interface. See Known Problems for specific instances of these and other problems.

Problems Which Affect Other Applications

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

- Accelerator Capture cursor won't disappear.
- Accelerator Menu command keys interfere with normal key use.
- Newly opened windows close prematurely, or fail to close normally.
- Owned windows get lost or float above their background.
- Z movement initiated by other applications, such as Always On Top, fails.

Capture problems can be solved by disabling the Capture Window sub-option in the Accelerator Menu Controls. Command key problems can be solved by disabling the Command Key sub-option in the Accelerator Menu Controls. Some problems concerning disappearing windows can be solved by holding down the Shift key and moving the mouse cursor onto the window. Others require that the Automatic Activation sub-option for main and/or MDI child windows in the Maintain Z Controls be disabled.

See Known Problems for issues concerning owned windows, pseudo-MDI applications, Always On Top, and specific instances of these and other problems.

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 corrected. Some the user can probably live with quite comfortably. Others the user must avoid or, trading WinX capability for safety, certain options and sub-options in WinX must be disabled.

General Problems

Always On Top

If the Maintain Z option in the Maintain Z Controls is enabled, Always On Top in other applications will not work immediately or consistently. Windows provides no way for WinX to distinguish between a window being moved to the "top", which it blocks, and a window being moved to Always On Top, which it should allow. The solution is to use the Always On Top command in the WinX Accelerator Menu exclusively, or to disable the Maintain Z option in the Maintain Z Controls.

Many applications which provide an Always On Top feature do not actually check the status of the target window--they assume that if Always On Top was set then it must still be set, and vice versa. If WinX is used to change the Always On Top status of the window then the application's Always On Top mechanism may be out of sync with reality, and indicate that the window is not Always On Top. The solution is to always use and reference Always On Top in the WinX Accelerator Menu since it checks and indicates a window's true status.

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 right hand side of the display, the menu will appear under the mouse cursor instead of above, below, or to the left of it. This often causes a menu item to be inadvertently selected and, when the mouse button is released, to be executed. Being unexpected, this action may appear to the user as some random bug in WinX.

With the Close Menus option of the Close Menus Control enabled, menus occasionally fail to close upon release of the mouse button. The problem is intermittent and seems to only occur on high speed (66MHz) systems when displaying long menus. In these cases, clicking outside the menu or hitting the Esc key closes the menu.

Owned Windows (Lost, Hidden and Floating)

With no options or sub-options in the Owner Behind Controls enabled, any window on the desktop can be moved in Z without restriction or affecting other windows. This sometimes can cause seemingly unrelated problems.

Some applications, such as for application setup, take over the screen. With WinX it is possible to push windows back behind this "blanket" window. Once a window is lost behind the blanket window there is often no way to retrieve it, and not much else can be done with the system except to reboot it. This problem can be solved by first restarting Windows, and then enabling the Owner Behind option in the Owner Behind Controls.

Some applications create a "window" by filling the main window with a background and painting the information in a separate overlying window which it owns. Sometimes, because of the order in which these windows are painted, the information window gets hidden behind its owner. In other cases it is possible for the information window to float above its background, with other windows coming in-between the two. The first case often can be solved by enabling the Owner Behind option in the Owner Behind Controls. The second case is often solved by enabling the Owned Together sub-option in the Owner Behind Controls.

Maximized MDI Children

If an MDI child window is maximized, activating a sibling MDI child window will cause the sibling to maximize and pop to the front. This is a feature of Windows, not WinX.

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 applications such as these appear to be Multiple Document Interface (MDI) applications, internally they are not. Since the activation process of windows belonging to such applications is private, WinX is unable to effect or detect the activation of their child windows.

Drop Down Lists

Lists of items which drop down when a button is clicked are implemented by applications in a variety of ways. In most applications it is possible to move the "parent" window forward in Z while a list is open, thereby covering up the list. Some lists are interpreted as main windows which, if the Capture Window sub-option in the Accelerator Menu Controls is enabled, get captured when they open. Some drop down lists are also considered to be "shy" windows (see below), which disappear when another window activates or the mouse is clicked.

Due to the variety of implementations involved, it is difficult if not impossible for WinX to handle all of them consistently and correctly. If the drop down list fails to appear or becomes covered, a temporary solution is to push the parent window backwards using the Accelerator Menu or command keys until the list is exposed. A more permanent solution is to disable the Maintain Z option in the Maintain Z Controls. For certain cases, enabling the Owner Behind option in the Owner Behind Controls can help prevent the list from being covered by its parent.

Shy Windows

A "shy" window is one which disappears when it deactivates. Examples are the Windows Task List, popup windows in Windows Help, and the Toolbox and Properties windows in the Visual C++ AppStudio. Shy windows are implemented by applications in a variety of ways. Some drop down lists (see above) also fall into this category. Such windows generally activate when they popup. Thus, losing Accelerator Capture by moving the mouse, or activating a different window by clicking on it causes activation to change and the shy window to disappear before it can be used.

A temporary solution is to hold down the Shift key, which prevents activation from changing, and then to move the mouse cursor onto the shy window. A permanent solution is to disable the Main Auto-activation sub-option in the Maintain Z Controls.

Minor Quirks

Normally, when the Task List pops up it also activates. The Task List, however, is a shy window (see above) which disappears when it deactivates. To keep it from disappearing when the Main Auto-activation sub-option in the Maintain Z Controls is enabled, WinX blocks the Task List activation when it pops up.

Normally, Windows pushes a minimized window to the back of the Z order. When the Maintain Z option in the Maintain Z Controls is enabled, a minimized window will maintain its Z position.

When an application, such as WinX, is run hidden at startup, and the Accelerator Capture sub-option in the Accelerator Menu Controls is enabled, the hidden window will be captured.

Application Specific Problems

Windows 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, and use of the Shift key to modify the action of options and sub-options (e.g. Size/Move Border Control). 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.

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

Windows Task List

When using the Task List, if the Cancel button is pushed, the previously active window will pop to the front and open even if the Maintain Z option in the Maintain Z Controls enabled. Windows provides no way for WinX to differentiate between a normal Task List selection and the "selection" generated by the Cancel button.

Windows Screen Saver

If Accelerator Capture is active when the Windows screen saver starts up, Accelerator Capture cursor will remain visible. Windows provides no way for WinX to detect when this occurs so that it can end capture and allow the cursor to disappear. This problem can be solved by disabling the Accelerator Capture sub-option in the Accelerator Menu Controls.

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

Windows File Manager

When dragging file or directory icons (objects) from one MDI Child window to another in the File Manager, the File Manager will sometimes complain that the file can not be found. This is because the File manager always assumes that the ***front*** window is the one ***from*** which the object is being moved! This problem can be solved by either disabling the Maintain Z option in the Maintain Z Controls, or by making sure that the MDI Child window from which the object is being dragged is in front.

Windows Help

When using the Always On Top item in the Help menu of a Windows Help window, the Help window will sometimes be filled with gray. An immediate solution is to toggle the window's Always On Top status with the WinX Accelerator Menu (popped up on the Help window border). A permanent solution is to either disable the Maintain Z option in the Maintain Z Controls, or to always using the WinX Accelerator Menu to control a window's Always On Top status.

Popup windows associated with Windows Help are often "shy" windows. See the section on **Shy Windows** above.

Microsoft 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. See the section on **Drop Down Lists** above. This drop down list is also a "shy" window. See the section on **Shy Windows** above.

Sometimes while debugging a program, and a break occurs, WinX is temporarily blocked from operating. Continuing execution of the program or stopping the debugging unblocks WinX.

Microsoft Excel

Microsoft Excel (all versions) appears to be an MDI application, but it is not. See the section on **Pseudo-MDI Applications** above.

With Excel 4.0 (and possibly other versions), if the Excel main window is active, the WinX Accelerator Menu can pop up but no menu selections can be made. Evidently Excel intercepts the message stream before WinX can access it with a hook. A solution is to use the Accelerator Menu command keys instead of the menu.

Microsoft Word

Microsoft Word (all versions) appears to be an MDI application, but it is not. See the section on **Pseudo-MDI Applications** above.

With Word 6.0, if a window is moved in Z and the cursor lands on the client area of the Word main window, the moved window will be captured but the capture cursor will not appear until the mouse is bumped or a key is pressed.

After Dark Screen Saver

Windows which are Always On Top will appear on top of the After Dark screen saver (and perhaps others). This is a feature of Windows and/or After Dark, not WinX.

Alt+Tab and Normalize in the Accelerator Menu fail to "open" the After Dark icon. This is because the After Dark control panel is not the open state of the icon. This has nothing to do with WinX. To make the control panel appear, double click on the icon or select Control Panel in its system menu.

WinDock

The WinDock shareware product can interfere with the operation of WinX. For example, Automatic Activation and Accelerator Capture fail to work. Evidently, WinDock uses system hooks put fails to chain them properly for other hook users. A temporary solution is to run WinDock before running WinX, or simply disable and then re-enable WinX with the Enable/Disable WinX Button. Both allow WinX to have access to the system hooks before WinDock.

Central Point PC Tools

The Desktop application uses a lot of "tricks" to do what it does in spite of Windows. As such, it deviates significantly from the norm as regards application implementation, and WinX has a hard time dealing with certain aspects of it.

If the Click Border sub-option in the Maintain Z Controls is enabled, clicking on the Desktop "title bar" fails to also pop the "menu bar" to the front. This is because of the manner in which the Desktop application manages their Z position. Since it is private, WinX is unable to directly control the situation. A temporary solution is to move the front window backwards with the Accelerator Menu or command keys until the menu bar is exposed. A permanent solution is to disable the Maintain Z option in the Maintain Z Controls.

In general, the Owned Together sub-option in the Owner Behind Controls must be ***disabled*** when using the Desktop. If not, then clicking on a window or the Desktop title bar or menu bar to pop them to the front causes a transparent "background" window to also move to the front. Once in front, the background window blocks most mouse clicks and causes a beep. A temporary solution is to popup the Accelerator Menu and select Close in order to kill the Desktop. Then, disable the Owned Together sub-option in the Owner Behind Controls and leave it disabled to prevent a re-occurrence. A permanent solution is to disable the Maintain Z option in the Maintain Z Controls.

Many of the "windows" in Desktop are not real windows. As such, WinX can not interact with them or can only interact with them in a limited way. The Desktop icons are not real icons. WinX can not activate them or move them in Z. The MultiDesk window is a modified window. Its borders are normal, but its title bar is not real. Thus, clicking on it does not pop the window to the front, and pressing the right mouse button on it does not popup an Accelerator Menu. When capture occurs on a non-real window or the background, the captured "window" is actually the Desktop application. Z movement is ignored, and selecting Close in the Accelerator Menu kills the Desktop application.

Compuserve WinCIM

If a window is covering the WinCIM main window, and certain conditions in WinCIM cause it to pop to the front, such as receiving mail, the icons in the tool bar often do not repaint and appear blank. A temporary solution is to minimize and then restore the main window. A permanent solution is to disable the Maintain Z option in the Maintain Z Controls.

See Also

Questions and Comments

Questions and Comments

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