

Windows

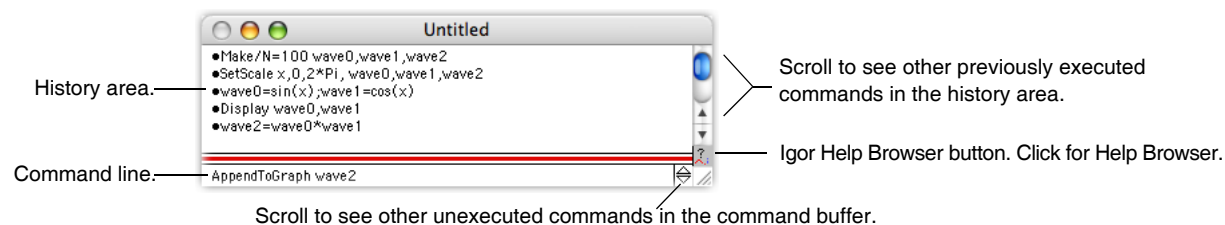
Overview	54
The Command Window	54
The Rest of the Windows.....	54
The Target Window	55
Window Names and Titles	56
Allowable Window Names	57
The Open File Submenu	58
The Windows Menu	58
Making a New Window.....	58
Activating Windows	58
Showing and Hiding Windows	58
Closing a Window	59
Killing Versus Hiding	59
Saving the Window Contents	59
Close Window Dialogs	60
Saving a Window as a Recreation Macro	61
Window Macros Submenus	62
The Name of a Recreated Window	63
Changing a Window's Style From a Macro.....	63
The Window Control Dialog.....	64
Arranging Windows.....	65
The Tile or Stack Windows Dialog.....	66
Window Position and Size Management	67
Move to Preferred Position	67
Move to Full Size Position.....	67
Retrieve Window	67
Retrieve All Windows.....	67
Send to Back — Bring to Front.....	67
Text Windows.....	68
Executing Commands	68
Text Window Navigation	68
Finding Text in the Active Window.....	69
Find and Replace	69
Finding Text in Multiple Windows.....	69
Text Magnification	71
Window User Data	72
Chapters About Specific Windows	72
Window Shortcuts	73

Overview

This chapter describes Igor's windows in general terms, just a bit about the File menu, and the Windows menu and window recreation macros in detail.

The Command Window

When Igor first starts, the **command window** appears at the bottom of the screen:



Commands are automatically entered and executed in the command window's **command line** when you use the mouse to "point-and-click" your way through dialogs. You may optionally type the commands directly and press Return or Enter. Igor preserves a history of executed commands in the **history area**.

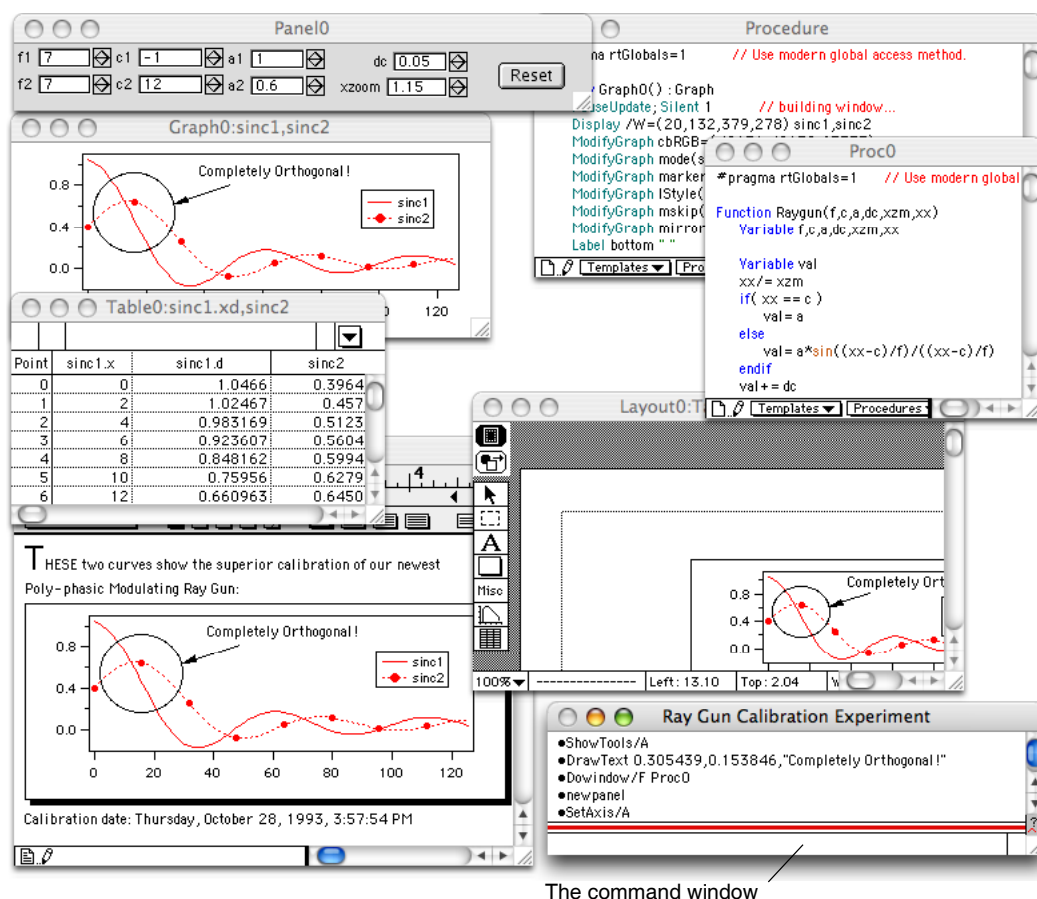
For more about the command window, see Chapter II-2, **The Command Window**, and Chapter IV-1, **Working with Commands**.

The Rest of the Windows

At startup, by default, Igor displays a table window. There are also a number of additional windows which are initially hidden:

- The main procedure window
- The Igor Help Browser
- Help windows for files in "Igor Pro Folder/Igor Help Files" and "Igor Pro User Files/Igor Help Files"

You can create additional windows for graphs, tables, page layouts, notebooks, panels and auxiliary procedure windows, as well as more help windows.



The command window

See the section **Chapters About Specific Windows** on page II-72.

Igor extensions may add other windows to Igor. For example, the Data Browser window, which lets you see what data exists in the current experiment, is added by the Data Browser extension.

The Target Window

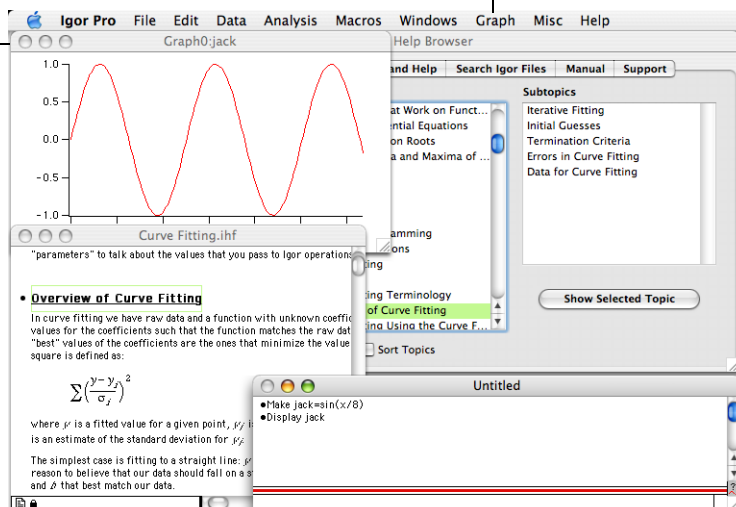
Igor commands and menus operate on the **target window**. The target window is the top graph, table, page layout, notebook, control panel or XOP target window. The term "target" comes from the fact that these windows can be the target of command line operations such as `ModifyGraph`, `ModifyTable` and so on. The command window, procedure windows, help windows and dialogs can not be targets of command line operations and thus are not target windows.

Prior to version 4, Igor attempted to draw a special icon to indicate which window was the target. However, this special target icon is no longer drawn because of operating system conflicts.

The menu bar changes depending on the top window and the target window. For instance, if a graph is the target window the menu bar contains the Graph menu:

Items in the Graph menu will affect Graph0.

Graph0 is the target window, even though the command window and the Using Igor help window are in front of it.



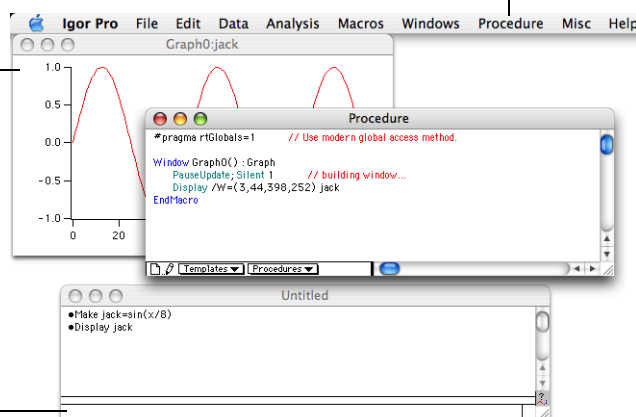
The menu bar changes to contain menus that apply to the target window, but you may type any command into the command line, including commands that do not apply to the target window. Igor will apply the command to the top window of the correct type.

For instance, in the example above, you could type a `ModifyTable` command while Graph0 was the target window. Igor will apply the `ModifyTable` command to the top table, if there is one.

Sometimes the top window isn't a target window, but it causes the menu bar to change. To continue our example, if at this point you were to bring a procedure window to the top, the graph would still be the target window, but the Graph menu would be replaced with the Procedure menu. Menu items chosen from the Procedure menu apply to the top procedure window, but typed commands like `AppendToGraph myWave` or `DoWindow` will still affect the target window, Graph0.

Items in the Procedure menu will affect the Procedure window.

Graph0 is the target window, even though the command window and the main procedure window are in front of it.



Commands typed here will affect Graph0, the target window.

Window Names and Titles

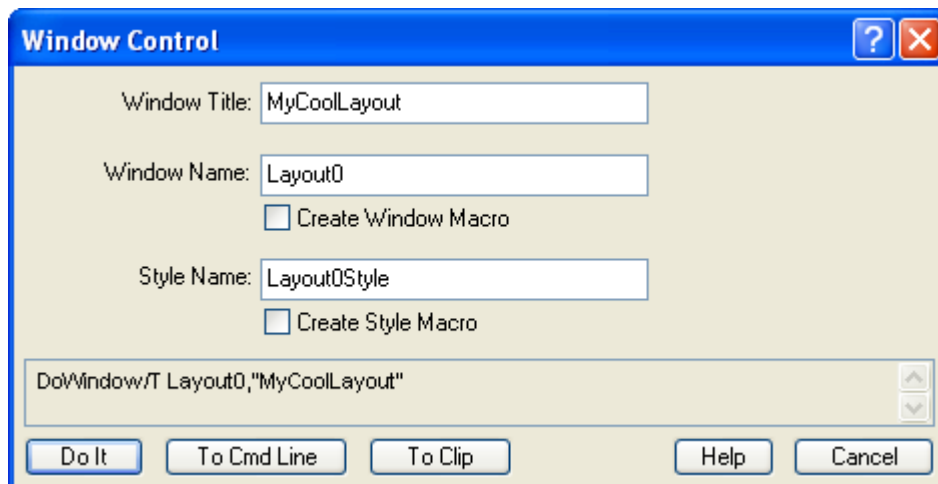
Each graph, table, page layout or panel has a **title** and a **name**.

The title is what you see at the top of the window frame and in the Windows menu. Its purpose is to help you visually identify the window, and is usually descriptive of its contents or purpose.

The window *name* is not the same as the *title*. The purpose of the name is to allow you to refer to the window from a command, such as the `DoWindow` or `AppendToGraph` operations.

When you first create one of these windows, Igor gives it a name like Graph0, Table0, Layout0 or Panel0, and a title based on the name and window contents. You can change the window's title and name to something more descriptive using the Window Control dialog (Windows→Control submenu). Among other things, it renames and retitles the target window.

Here we are about to change the title of the window named Layout0:



The Window Control dialog is also a good way to discover the name of the top window, since the window shows only the window title.

The command window, procedure windows, and help windows have *only* a title. The title is the name of the file in which they are stored. These windows do not have names because they can not be affected by command line operations.

In summary, you set the title of windows in various ways:

Window Type	How Titled	Has Window Name?
Graphs, tables, page layouts, notebooks and panels	Igor initially assigns a title based on the window name and content. You can retitle these windows with the Window Control dialog or the DoWindow/T command.	Yes
Command window	Initially "Untitled", it takes on the file name of saved experiment.	No
Built-in procedure window	Always titled "Procedure".	No
Auxiliary procedure windows	Titled when created, they take on the file name if saved as a file.	No
Igor Help windows	Same as the Igor help file.	No

Allowable Window Names

A window name is used for commands and therefore must follow the standard rules for naming Igor objects:

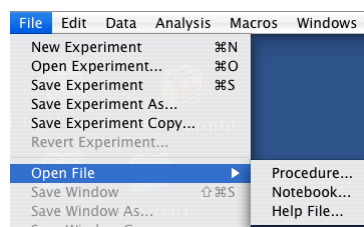
- The name must start with a letter.
- Additional characters can be alphanumeric or the underscore character.
- No other characters, including spaces, are allowed in standard Igor object names.
- No more than 31 characters are allowed.
- The name must not conflict with other object names (you see a message if it does).

For more information, see **Object Names** on page III-415.

The Open File Submenu

The File menu contains the Open File submenu for opening an existing file as a notebook, Igor help window, or procedure window.

When you choose an item from the submenu, the Open File dialog appears for you to select a file.



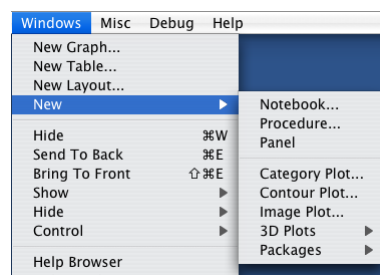
The Windows Menu

You can use the Windows menu for making new windows, and for showing, arranging and closing (either hiding or “killing”) windows. You can also execute “window recreation macros” that recreate windows that have been killed and “style macros” that modify an existing window’s appearance.

Making a New Window

You can use the various items in the Windows→New submenu to create new windows.

The menu items that end with “...” invoke dialogs which produce commands that Igor executes to create the windows. These dialogs are explained in the chapter about the corresponding window.



You can type these commands yourself directly in the command line. For example,

```
Display yData vs xData
```

creates a graph of the wave named yData on the Y axis, versus xData on the X axis.

You can create a new window by selecting the name of a window recreation macro from the Windows menu. See **Window Macros Submenus** on page II-62.

You can also create a window using the File→Open File submenu.

Activating Windows

To activate a window, choose an item from Windows menu or from the Help Windows, Procedure Windows, Graphs, Tables, Layouts, Other Windows, or Recent Windows submenus in the Windows menu.

The Recent Windows submenu shows windows recently activated. This information is saved when you save an experiment to disk and restored when you later reopen the experiment.

If you press Command (*Macintosh*) or Ctrl (*Windows*) while clicking the menu bar, a temporary Recent Windows menu will be accessible from the main menu bar. This shortcut is intended to save you the trouble of navigating through the Windows menu to the permanent Recent Windows submenu.

By default, just the window’s title is displayed in the Windows menu. You can choose to display the title or the name for target windows using the Windows Menu Shows popup menu in the Misc Settings category of the Miscellaneous Settings dialog.

Showing and Hiding Windows

All built-in window types and some XOP window types can be hidden.

To hide a window, press Shift and choose Windows→Hide or use the keyboard shortcut Command-Shift-W (*Macintosh*) or Ctrl+Shift+W (*Windows*). You can also hide a window by pressing Shift and clicking the close button.

You can hide multiple windows at once using the Windows→Hide submenu. For example, to hide all graphs, choose Windows→Hide→All Graphs. If you press Shift while clicking the Windows menu, the sense of the menu items changes. For example, Hide→All Graphs changes to Hide→All Except Graphs.

The command window is not included in mass hides of any kind. If you want to hide it you must do so manually.

Similarly, you can show multiple windows at once using the Windows→Show submenu. For example, to show all graphs, choose Windows→Show→All Graphs. If you press Shift while clicking the Windows menu, the sense of the menu items changes. For example, Show→All Graphs changes to Show→All Except Graphs.

The Show All Except menu items do not show procedure windows and help files because there are so many of them that it would be counterproductive.

The Windows→Show→Recently Hidden Windows item shows windows recently hidden by a mass hide operation, such as Hide→All Graphs, or windows recently hidden manually (one-at-a-time using the close button or Command-Shift-W or Ctrl+Shift+W). In the case of manually hidden windows, “recently hidden” means within the last 30 seconds.

XOP windows do participate in Hide All XOP Windows and Show All XOP Windows only if XOP programmers specifically support these features.

Closing a Window

You can close a window by either choosing the Close item or by clicking in the window’s close button. Depending on the top window’s type, this will either kill or hide the window, possibly after a dialog asking for confirmation.

Killing Versus Hiding

“Killing” a window means the window is removed from the experiment. The memory used by the window is released and available for other purposes. The window’s title is removed from the Windows menu. Killing a window that represents a file on disk does not delete the file. You can also kill a window with a `DoWindow/K winName` command.

“Hiding” a window simply means the window is made invisible, but is still part of the experiment and uses the same amount of memory. It can be made visible again by choosing its title from the Windows menu.

The command window and the built-in procedure window can be hidden but not killed. All other built-in windows can be hidden or killed.

When you create a window from a procedure, you can control what happens when the user clicks the close button using the `/K=<num>` flag in the command that creates the window.

You can hide a window programmatically using the `DoWindow/HIDE=1` operation. To show a hidden window without activating it, use `DoWindow/HIDE=0`. To show the window and activate it, use `DoWindow/F`.

Saving the Window Contents

Notebooks and procedure windows can be saved either in their own file, or in a packed experiment file with everything else. You can tell which is the case by choosing Notebook→Info or Procedure→Info. When you kill a notebook or a procedure window that contains unsaved information, a dialog will allow you to save it before killing the window.

Graph, table, panel and page layout windows are not saved as separate files, and are lost when you kill them unless you save a **window recreation macro** which you can execute to later recreate the window. Killing these windows and saving them as window recreation macros (stored in the built-in procedure

Chapter II-4 — Windows

window) frees up memory and reduces window clutter without losing any information. You can think of window recreation macros as “freeze-dried windows”.

This table shows how windows are hidden, killed, and saved:

Window Type	Hideable?	Killable?	Save Recreation Macro?	Stand-Alone File?
Graphs, Tables	Yes	Yes	Yes	Yes*
Layouts, Panels	Yes	Yes	Yes	No
Main procedure window	Yes	No	No	†
Help Browser	Yes	No	No	No
Auxiliary procedures, Notebooks, Help windows	Yes	Yes‡	No	Yes
Command window	No	No	No	No

* The **SaveGraphCopy** operation (page V-539) and the **SaveTableCopy** operation (page V-546) can be used to save a graph or table, along with all associated waves, as a stand-alone experiment file.

† The main procedure window is stored as a separate file in the home folder if the experiment is saved in unpacked format, or within the experiment file if the experiment is saved in packed format.

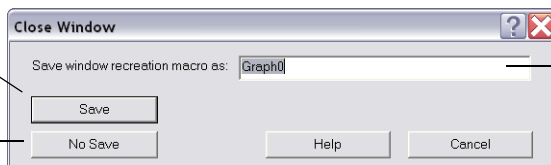
‡ The only way to kill a help window is to press Option (*Macintosh*) or Alt (*Windows*) when clicking the close button, or by pressing Command-Option-W (*Macintosh*) or Ctrl+Alt+W (*Windows*).

Close Window Dialogs

When you close a graph, table, layout or control panel, Igor presents a Close dialog.

Saves (or replaces) a recreation macro and kills the window.

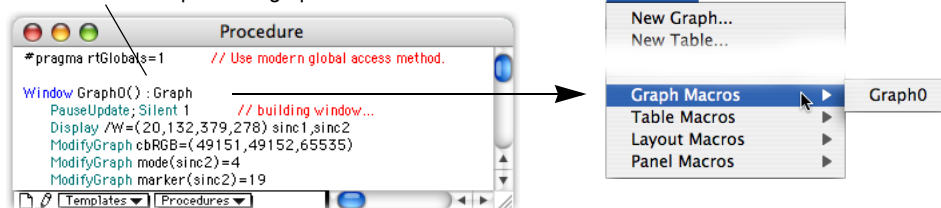
Kills the window without saving a recreation macro.



Name of the recreation macro.

If you click the Save button Igor creates a window recreation macro in the main procedure window. It sets the macro's subtype to Graph, Table, Layout or Panel so the name of the macro appears in the appropriate Macros submenu of the Windows menu. You can recreate the window using this menu.

:Graph subtype identifies window recreation macro named Graph0 as a graph macro.



If you don't plan to use the window again, you should click the No Save button and no window recreation macro will be created.

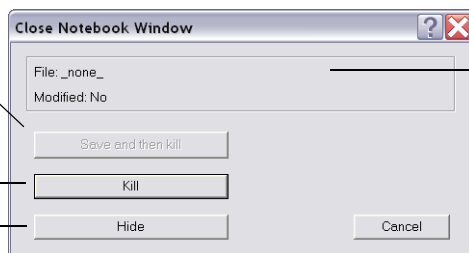
If you have previously created a recreation macro for the window then the dialog will have a Replace button instead of a Save button. Clicking Replace replaces the old window recreation macro with a new one. If you know that you won't need to recreate the window, you can delete the macro (see **Saving a Window as a Recreation Macro** on page II-61).

When you close a notebook or procedure window (other than the built-in procedure window), Igor presents a “hide or kill dialog”.

Saves the file (if any) and removes window from the experiment.

Removes the window from the experiment without saving.

Just hides the window.



The Notebook contents are stored in the experiment file, not in a separate notebook file.

Macintosh: When the Close Window dialog is showing, you can press Option to make the Kill button the default. The Kill button will become highlighted while the "Save and then kill" button will become normal. You can then press Return or Enter to kill the window. Similarly, press Shift to make the Hide button the default button.

To hide a window, press Shift while clicking the close button. To kill a graph, table, layout, or control panel without the Close dialog, press Option (*Macintosh*) or Alt (*Windows*) while clicking the close button.

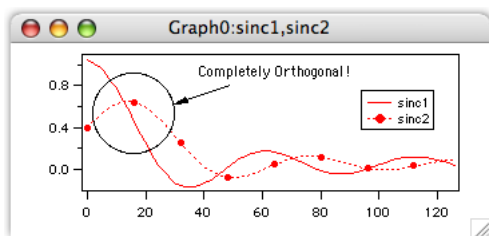
By specifying /K=<num> for the NewNotebook, Layout, Display, and NewPanel operations, you can modify this behavior.

Saving a Window as a Recreation Macro

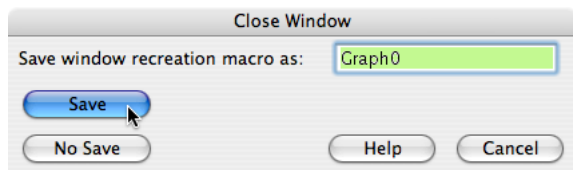
When you close a window that can be saved with a recreation macro, Igor offers to create one by displaying the Close Window dialog. Igor stores the window recreation macro in the main procedure window of the current experiment. The macro uses much less memory than the window, and reduces window clutter. You can invoke the window recreation macro later to recreate the window. You can also create or update a macro with the Window Control dialog.

The window macro contains all the necessary commands to reconstruct the window provided the underlying data is still present. For instance, a graph recreation macro contains commands to append waves to the graph, but does not contain any wave data. Similarly, a page layout recreation macro does not contain graphs or tables (nor the commands to create them). The macros refer to waves, graphs and tables in the current experiment by name.

Here is how you would use recreation macros to keep a graph handy, but out of your way:

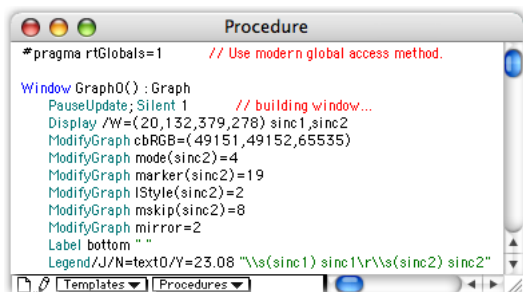


Choosing Close or clicking the close button...

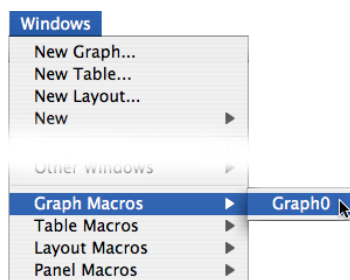


...summons the Close Window dialog.

Clicking Save...



...saves a window recreation macro for the graph in the main procedure window.



When the graph window is needed again choosing the macro from the Graph Macros menu runs the macro that recreates the graph.

The window macro is evaluated in the context of the root data folder. This detail is of consequence only to programmers. See **Data Folders and Commands** on page II-125 for more information.

You can create or replace a window macro without killing the window using **The Window Control Dialog** described on page II-64. The most common reason to replace a window macro is to keep the macro consistent with the window that it creates. This is useful if you are about to clone the window, having changed it since the recreation macro was made.

Notice that the proposed name of the window recreation macro is the same as the name of the saved window. You can save the window recreation macro under a different name, if you want, by entering the new name in the dialog. If you do this, Igor creates a new macro and leaves the original macro intact. You can run the new macro to create a new version of the window or you can run the old macro to recreate the old version. This way you can save several versions of a window, while displaying only the most recent one.

Window recreation macros stay in an experiment's procedure window indefinitely. If you know that you won't need to recreate a window for which a window recreation macro exists, you can delete the macro.

To locate a window macro quickly:

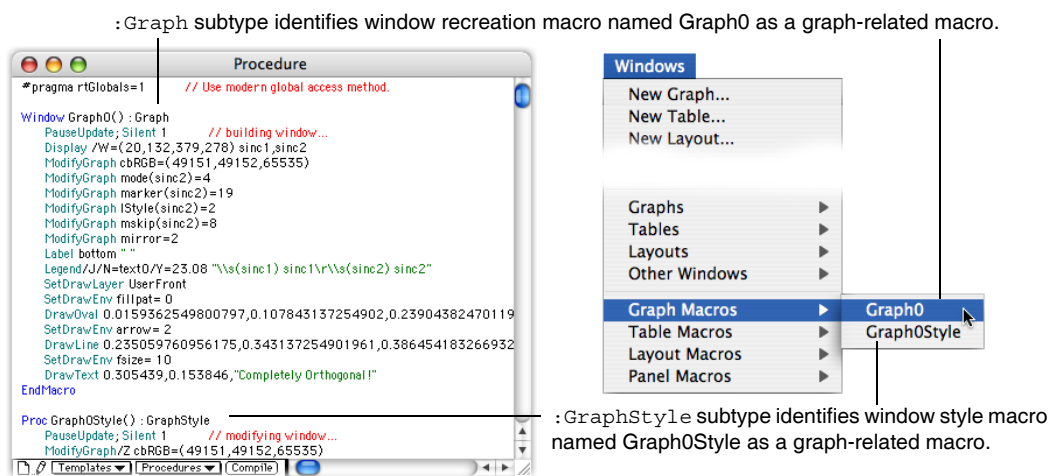
- Bring any procedure window to the top, press Option (*Macintosh*) or Alt (*Windows*) and choose the window macro name from the appropriate macro submenu in the Windows menu.

To delete the macro (if you're sure you won't want it again), simply select all the text from the Macro declaration line to the End line. Press Delete to remove the selected text.

See **Saving and Recreating Graphs** on page II-300 for details specific to graphs.

Window Macros Submenus

The Windows menu has hierarchical menus containing graph, table, page layout and panel recreation macros. These menus also include graph, table or page layout style macros.



Window recreation macros are created by the Close Window and Window Control dialogs, and by the DoWindow/R command. Style macros are created by the Window Control dialog and the DoWindow/R/S command.

Igor places macros into the appropriate macro submenu by examining the macro's subtype. The subtypes are Graph, Table, Layout, Panel, GraphStyle, TableStyle and LayoutStyle. See **Procedure Subtypes** on page IV-179 for details.

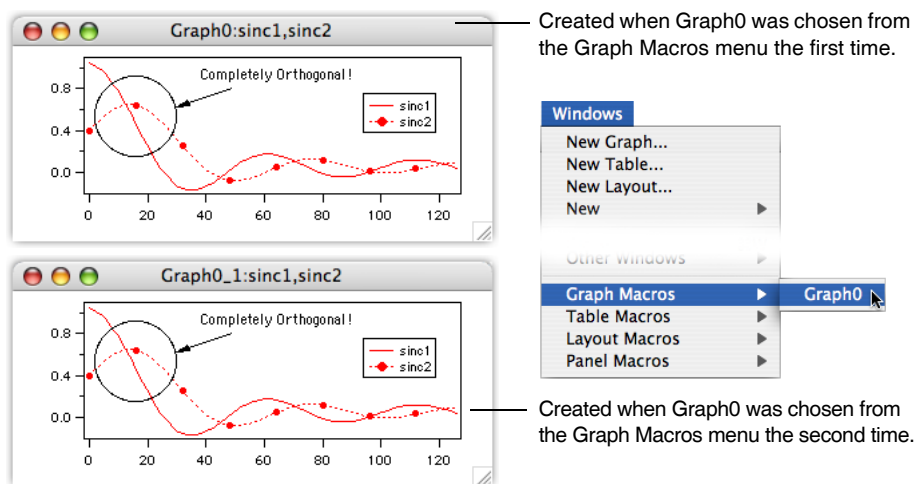
When you choose the name of a recreation macro from a macro submenu, the macro runs and recreates the window. Choosing a style macro runs the macro which changes the target window's appearance (its "style").

However, if a procedure window is the top window and you press Option (*Macintosh*) or Alt (*Windows*) and then choose the name of any macro, Igor displays that macro but does not execute it.

The Name of a Recreated Window

When you run a window recreation macro, Igor recreates the window with the same name as the macro that created it unless there is already a window by that name. In this case, Igor adds an underscore followed by a digit (e.g. _1) to the name of the newly created window to distinguish it from the preexisting window.

For example, this figure shows the result of running a graph recreation macro twice. There was no graph named Graph0 when we started:



Changing a Window's Style From a Macro

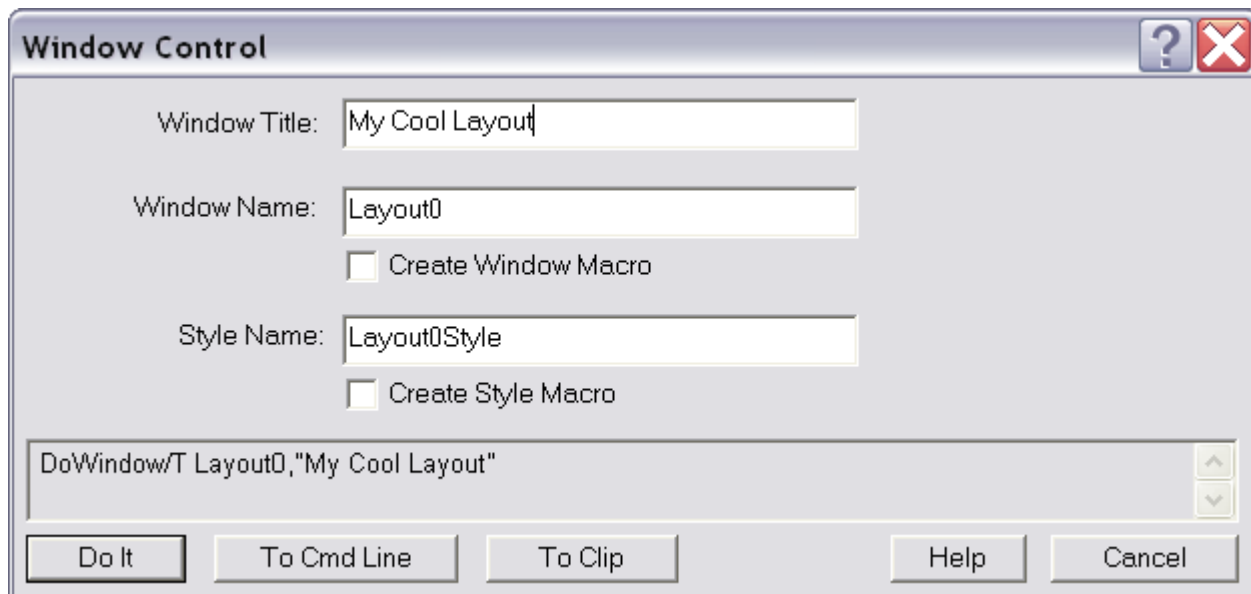
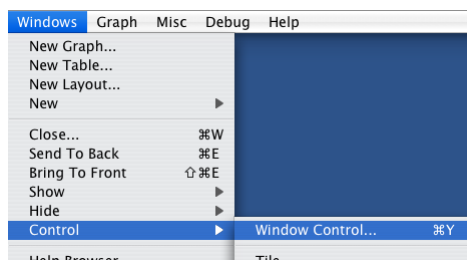
When you run a style macro by invoking it from the Windows menu, from the command line or from another macro, Igor applies the commands in the macro to the top window. Usually these commands change the appearance of the window. For example, a graph style macro may change the color of graph traces or the axis tick marks.

Style macros are used most effectively with graph windows. For more information, see **Saving and Recreating Graphs** on page II-300 and **Graph Style Macros** on page II-300.

The Window Control Dialog

Choosing Control→Window Control brings up a dialog you can use to change the top window's title and name, and create or update its recreation and style macros. You can access this dialog quickly by pressing Command-Y (*Macintosh*) or Ctrl+Y (*Windows*).

Here we are using the dialog to change the window named Layout0 to have a new title of "My Cool Layout".

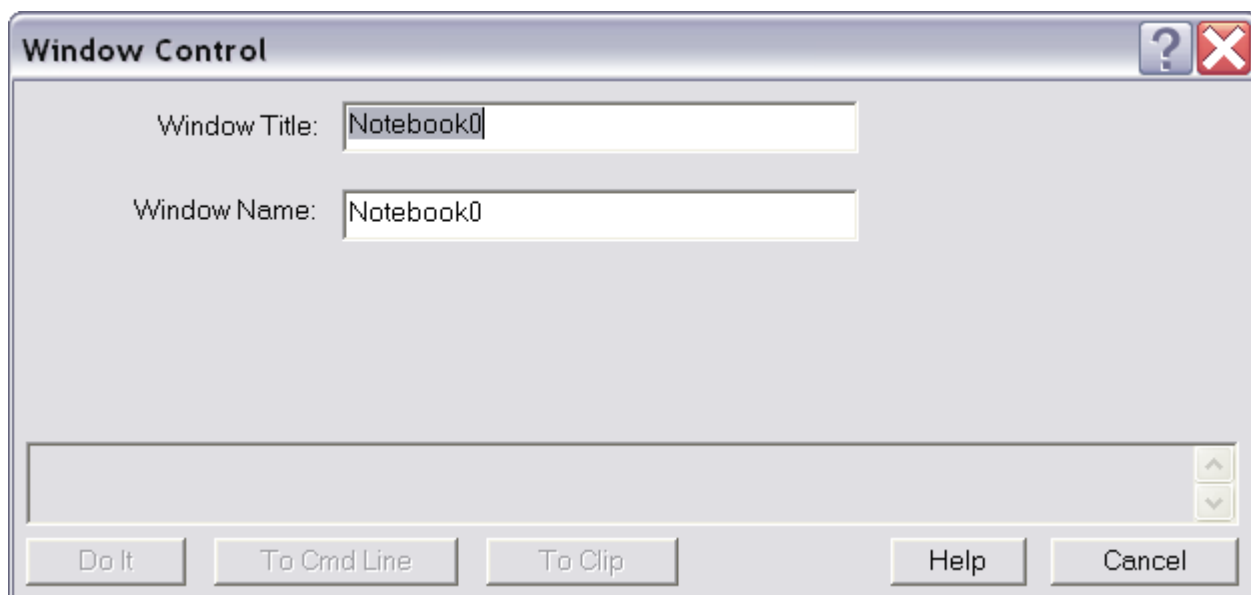


You can also change the window's name. The window name is used to address the window from command line operations such as DoWindow and also appears in the macro submenus of the Windows menu.

If the window name matches the name of an existing window or style macro, the checkboxes will change to Update Window Macro and Update Style Macro.

The dialog may look a little different for some window types. For instance, panels don't have style macros, so for panel windows the Create Style Macro item will be missing.

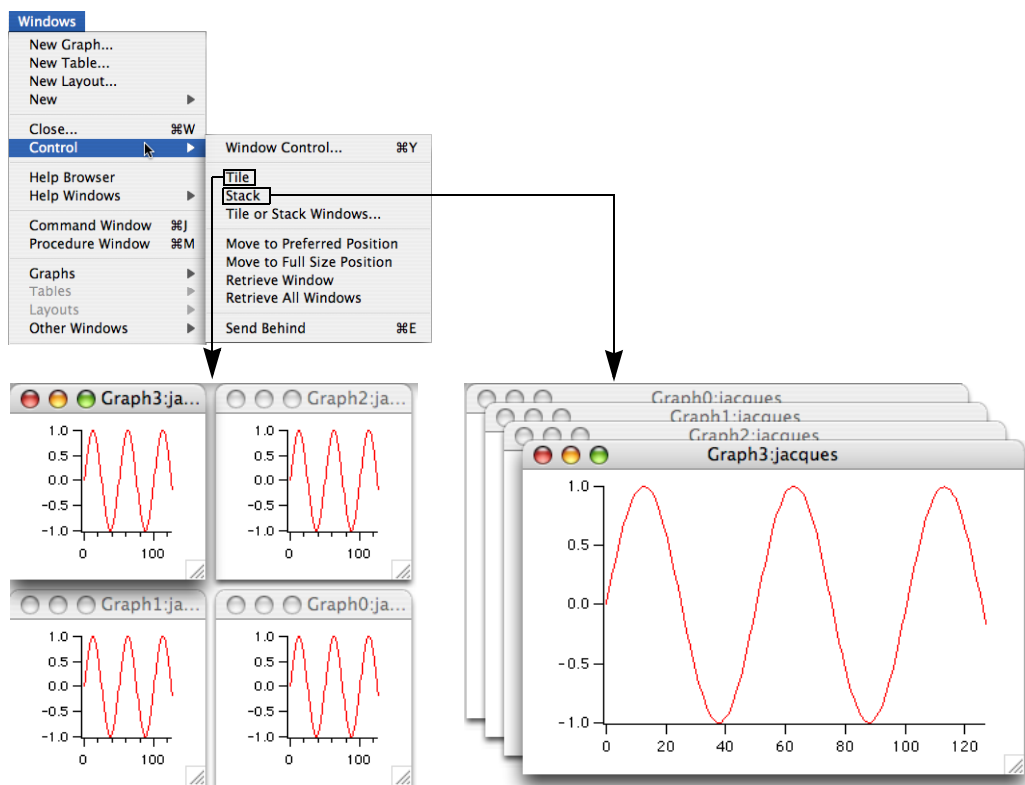
Similarly, notebooks can not be saved as macros, so both the Create Window Macro and Create Style Macro items will be missing:



For more about names and titles, see **Window Names and Titles** on page II-56. Also see **Saving a Window as a Recreation Macro** on page II-61 for a discussion of window recreation macros, and see **Graph Style Macros** on page II-300 for details on style macros.

Arranging Windows

You can tile or stack windows by choosing the appropriate items from the Control submenu in the Windows menu.

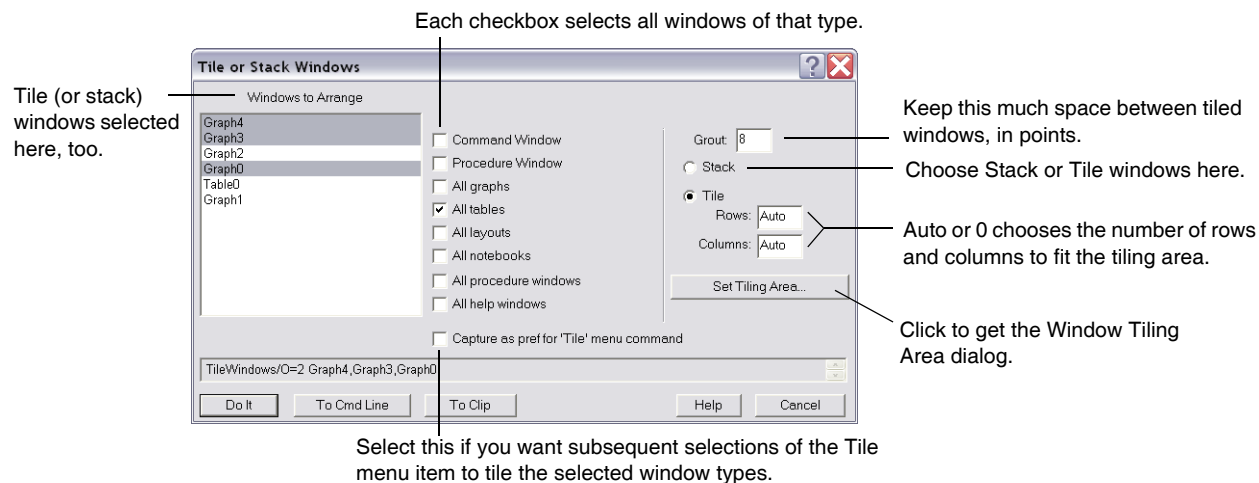


You can customize the behavior of the Tile and Stack items using the Tile or Stack Windows dialog.

You can also move windows around using the MoveWindow, StackWindows, and TileWindows commands.

The Tile or Stack Windows Dialog

The Tile or Stack Windows dialog is useful for tiling a few windows or even for setting the size and position of a single window.



Select individual windows from the Windows to Arrange list, and entire classes of windows with the checkboxes.

If you want subsequent selections of the Tile (or Stack) menu item to stack the same types of windows with the same rows, columns, grout, tiling area, etc., you should select the "Capture as pref" checkbox. Windows selected in the Windows to Arrange menu aren't remembered by the preferences: only the window type checkboxes. There are separate settings and preferences for Stack and for Tile.

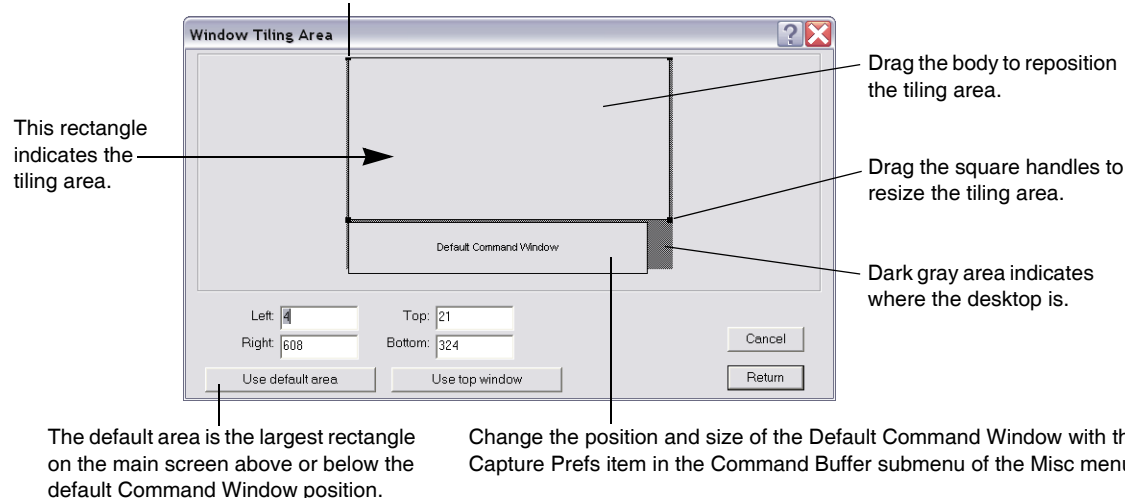
Notice that although the TileWindows and StackWindows operations can tile and stack panels, panels don't show up here because they don't resize very well.

The Window Tiling Area subdialog specifies the area where tiling *and stacking* take place.

You can specify the tiling area in one of four ways:

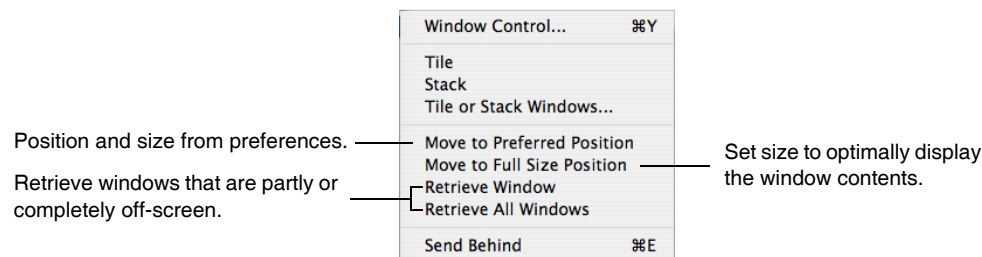
- By entering screen positions in units of points.
- By dragging the pictorial representation of the tiling area.
- Use the default tiling area by clicking the "Use default area" button.
- By positioning any nondialog window *before* you enter the dialog, and clicking the "Use top window" button.

The top, left corner has coordinates of 0,0.



Window Position and Size Management

There are four items in the Control submenu of the Windows menu that help you manage the position and size of windows.



Move to Preferred Position

Moves the active window to the position and size determined by preferences. For each type of window, you can set the preferred position and size using the Capture Prefs dialog (e.g., Capture Graph Prefs for graphs).

Shortcut for *Windows*: Press Alt and click the maximize button.

Move to Full Size Position

Moves and sizes the active window to display as much of the content as practical. On Macintosh, this is the same as clicking the zoom button. On *Windows*, the size is limited to the size of the frame window.

Shortcut: Press Shift-Option (*Macintosh*) or Shift+Alt (*Windows*) and click the maximize button.

Retrieve Window

Moves the active window and sizes it if necessary so that all of the window is visible.

Retrieve All Windows

Moves all windows and sizes them if necessary so that all of each window is within the screen on Macintosh or within the frame on *Windows*. This is often useful when you open an experiment that was created on a system with a larger screen or *Windows* frame than yours.

Send to Back — Bring to Front

The Send to Back item in the Windows menu sends the top window to the bottom of the desktop, behind all other windows. This function can also be accessed by pressing Control-Command-E (*Macintosh*) or Ctrl+E (*Windows*). After sending a window behind, you can bring it to the front by choosing Bring to Front or by pressing Shift-Control-Command-E (*Macintosh*) or Ctrl+Shift+E (*Windows*). You can also press Control-Command-E or Ctrl+E repeatedly to cycle through all windows.

Send to Back is handy to use in conjunction with Command-J (*Macintosh*) or Ctrl+J (*Windows*) which brings the command window to the top of the desktop. You can press Command-J or Ctrl+J to bring the command window to the top, enter a command, and then press Control-Command-E or Ctrl+E to get the command window out of the way again.

Igor has a nifty feature that comes in handy if you have many windows tiled such that some are completely behind others. If you press Option (*Macintosh*) or Alt (*Windows*) and choose Send to Back or press Command-Option-E (*Macintosh*) or Ctrl+Alt+E (*Windows*), any window that is completely visible is sent to the back.

For example, imagine that you have eight graphs. You can tile them into two planes of four graphs per plane using the Tile or Stack Windows dialog, or with the command: `TileWindows/O=1/A=(2,2)`. Now, pressing Command-Option-E or Ctrl+Alt+E sends the top four graphs behind, revealing the bottom four graphs.

You can also send a window to the back with the `DoWindow/B` command and bring it to the front with the `DoWindow/F` command.

Text Windows

Igor Pro displays text in procedure, notebook, and Igor help windows as well as in the command and history areas of the command window. This section discusses behavior common to all of these windows.

Executing Commands

You can execute commands selected in a notebook, procedure or help window by pressing Control-Enter or Control-Return. You can also execute selected commands by Control-clicking (*Macintosh*) or right-clicking (*Windows*) and choosing Execute Selection.

For more on this, see **Notebooks as Worksheets** on page III-5.

Text Window Navigation

The term “keyboard navigation” refers to selection and scrolling actions that Igor performs in response to the arrow keys and to the Home, End, Page Up, and Page Down keys. Macintosh and Windows have different conventions for these actions in windows containing text. You can use either Macintosh or Windows conventions on either platform.

By default, Igor uses Macintosh conventions on Macintosh and Windows conventions on Windows. You can change this using the Keyboard Navigation menu in the Misc Settings section of the Miscellaneous Settings dialog. If you use Macintosh conventions on Windows, use Ctrl in place of the Macintosh Command key. If you use Windows conventions on Macintosh, use Command in place of the Windows Ctrl key.

Macintosh Text Window Navigation

Key	No Modifier	Option	Command
Left Arrow	Move selection left one character	Move selection left one word	Move selection to start of line
Right Arrow	Move selection right one character	Move selection right one word	Move selection to end of line
Up Arrow	Move selection up one line	Move selection up one paragraph	Move selection up one screen
Down Arrow	Move selection down one line	Move selection down one paragraph	Move selection down one screen
Home	Scroll to start of document	Scroll to start of document	Not used
End	Scroll to end of document	Scroll to end of document	Not used
Page Up	Scroll up one screen	Scroll up one screen	Not used
Page Down	Scroll down one screen	Scroll down one screen	Not used

Windows Text Window Navigation

Key	No Modifier	Ctrl
Left Arrow	Move selection left one character	Move selection left one word
Right Arrow	Move selection right one character	Move selection right one word
Up Arrow	Move selection up one line	Move selection up one paragraph
Down Arrow	Move selection down one line	Move selection down one paragraph
Home	Move selection to start of line	Move selection to start of document
End	Move selection to end of line	Move selection to end of document

Windows Text Window Navigation

Key	No Modifier	Ctrl
Page Up	Scroll up 1 screen	Scroll up 1 screen
Page Down	Scroll down 1 screen	Scroll down 1 screen

Finding Text in the Active Window

You can access the Find Text dialog via the Edit menu or by pressing Command-F (*Macintosh*) or Ctrl+F (*Windows*). The Find Text dialog is available for help, procedure, and notebook windows, for the command line and the history area, and for some XOP windows.

You can search for the next occurrence of a string (Edit→Find Selection) without using the dialog by selecting the string and pressing Command-Control-H (*Macintosh*) or Ctrl+H (*Windows*). (Unreformed old-timers can change the Macintosh key combination to its original setting of Command-H using the Miscellaneous Settings dialog.)

After doing a find, you can search for the same text again by pressing Command-G (*Macintosh*) or Ctrl+G (*Windows*) (Find Same in the Edit menu). You can search for the same text but in the reverse direction by pressing Command-Shift-G (*Macintosh*) or Ctrl+Shift+G (*Windows*).

You can abort a find by clicking the Stop button in the Find Text dialog or by pressing Command-period (*Macintosh*) or Ctrl+Break (*Windows*).

Find and Replace

To find and replace:

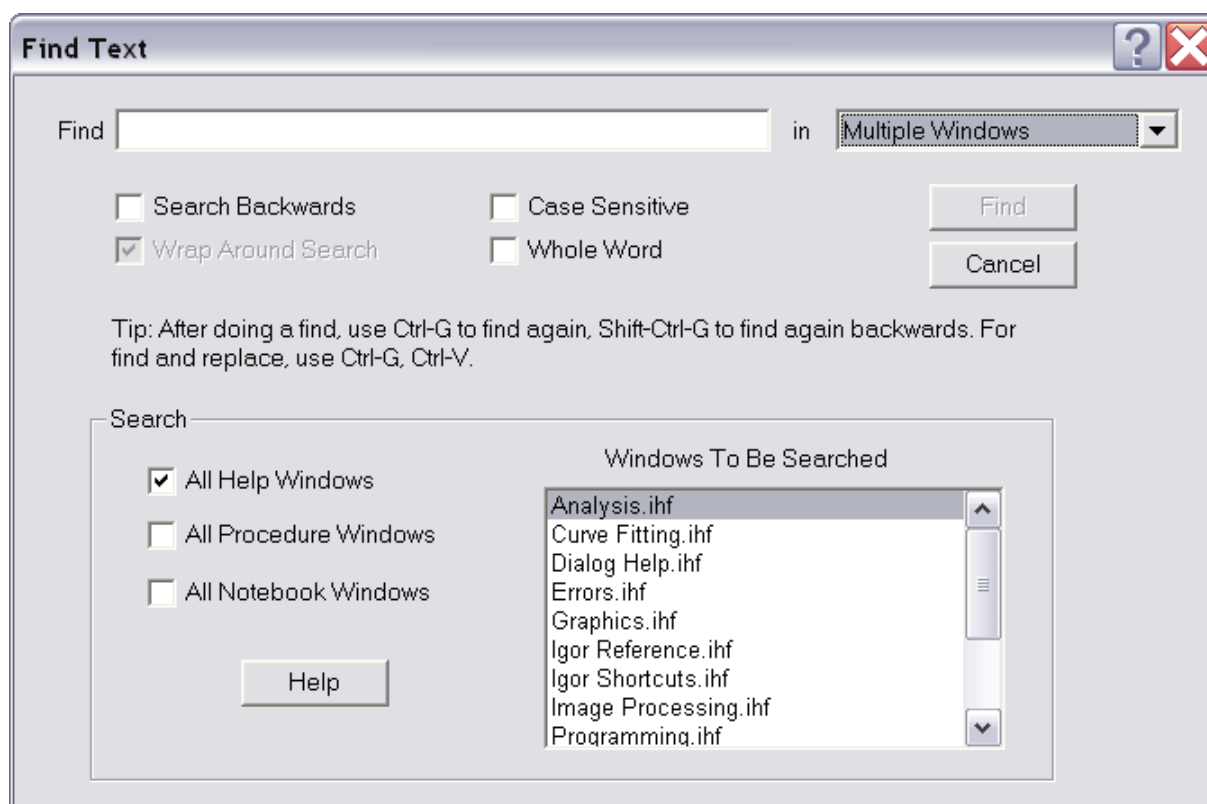
1. Move the selection to the top of the active window.
2. Use Edit→Find to find the first instance of the target string.
3. Manually change the first instance, then copy the new text to the Clipboard.
4. Press Command-G (*Macintosh*) or Ctrl-G (*Windows*) to find the next occurrence.
5. Press Command-V (*Macintosh*) or Ctrl-V (*Windows*) to paste.
6. Repeat steps 4 and 5 until done.

Finding Text in Multiple Windows

You can perform a Find on multiple help, procedure and notebook windows at one time using the following procedure.

1. Open a help, procedure or notebook window.
2. Press Command-F (*Macintosh*) or Ctrl+F (*Windows*) or choose Find from the Edit menu.
3. Choose Multiple Windows from the pop-up menu in the Find dialog.
4. Enter the text to find and click Find.

When you follow this procedure, the Find Text dialog looks something like this:



The windows that will be searched appear in the list, in the order in which they will be searched. You add windows to the list by selecting one of the checkboxes to the left of the list. The search is done from the top to the bottom of the list, or from the bottom to top if you have selected Search Backwards.

The selection in the list indicates the file to be searched next. You can change the selection by clicking the list or using the arrow keys. However, this is usually not necessary.

You can abort a find by clicking the Stop button in the Find Text dialog or by pressing Command-period (*Macintosh*) or Ctrl+Break (*Windows*).

When you turn multiple window find on, it stays on until you turn it off, by choosing Active Window Only from the pop-up menu. The setting of this pop-up menu affects not only Find but also Find Same and Find Selection. These last two operations do not display the Find Text dialog. To avoid confusion, use Find if you are unsure whether multiple window find is off or on.

The multiple window find may sometimes cause surprising behavior. For example, you may expect that the search will start with the active window. However, when doing a multiple window find, this is not the case. The search starts with the item highlighted in the list. Also, the search does not start from the selection in that window but rather from the top of the window, or from the bottom if you have selected Search Backwards.

When you click Find, Igor searches the windows in the list. When it finds the first occurrence of the target string, it closes the dialog and displays the found text. At this point, you can do a Command-G (*Macintosh*) or Ctrl+G (*Windows*) or choose Find Same from the Edit menu, to find the next occurrence of the target string. When you do a Find Same, the search starts from the selection in the window in which the target string was last found. This will normally be the active window but not if you activated another window after the last find.

While you are doing a multiple-window find you can, as in a single window find, press Command-Shift-G (*Macintosh*) or Ctrl+Shift+G (*Windows*) to search in the opposite direction (e.g., backwards if you were searching forwards). This provides a handy way to move quickly back and forth between two occurrences of the search string.

If you do a find after finding the last occurrence of the search string, Igor will beep, indicating that there are no more occurrences. At this point, if you want to go back to the preceding occurrence, you may need to press Command-Shift-G or Ctrl+Shift+G twice. The reason for this is that, after hitting the end of the list of windows to be searched, when you search backwards, Igor starts from the end of the list and finds the last occurrence again.

If you open or kill a window of a type that is to be searched, Igor rebuilds the list of files to be searched and resets the multiple-window find to the top of the list (or bottom if you are searching backwards). Igor also resets the multiple-window find when you change any of the settings in the Find Text dialog.

If you choose Find Selection from the Edit menu or press Command-H or Ctrl+H, Igor resets the find mode to active window only, enters the selected text as the search string, and then does the find. If what you really wanted was to do a multiple-window find, then after doing the Find Selection, press Command-F or Ctrl+F to active the Find Text dialog, choose Multiple Windows from the pop-up menu, and then click the Find button. This will start the multiple window find from the start of the list.

You can use the Igor Help Browser to search in multiple files, including files that are not open in your current experiment. See **Igor Help Browser** on page II-6 for further details.

Text Magnification

You can magnify the text in any window to make it bigger or smaller to suit your taste.

In help windows, procedure windows, plain text notebooks, and formatted text notebooks, you can use the magnifying glass icon in the bottom-left corner of the window. You can also use the Magnification submenu in the contextual menu for the window. To display the contextual menu, Control-click (*Macintosh*) or right-click (*Windows*) in the body of the window.

You can also set the magnification for the command line, history area, and text areas in dialogs such as Browse Waves and Add Annotation. These areas do not display the magnifying glass icon so you must use the contextual menu.

You may notice some anomalies when you use text magnification. For example, in a formatted text notebook, text may wrap at a different point in the paragraph and may change in relation to tab stops. This happens because fonts are not available in fractional sizes and because the actual width of text does not scale linearly with font size.

The Fit Width and Fit Page modes are inaccurate because of the availability of integer font sizes only. However they still may be useful. These modes are based on the available space for printing the document, which depends on the paper size chosen in the Page Setup dialog and the page margins as set in the Document Settings dialog. Because the actual content of the document may be much narrower or much wider than the available space for printing, these modes may sometimes give unexpected results.

You can set the default magnification for each type of text area by choosing a magnification from the Magnification popup menu and then choosing Set As Default from the same popup menu. Any text areas whose magnification is set to Default will use the newly specified default magnification. For example, if you want text in all help files to appear larger, open any help file, choose a larger magnification, 125% for example, and then choose Set As Default For Help Files. All help files whose current magnification is set to Default will be updated to use the new default.

The default magnification for the command line and history area controls the magnification that will be used the next time you launch Igor Pro.

The magnification setting is saved in formatted notebooks and help files only. If you change the magnification setting for one of these files and then save and close the file, the magnification setting will be restored when you reopen the file. For all other types of text areas, including procedure windows and plain text notebooks, the magnification setting is not stored in the file. If you close and reopen such a file, it will reopen using the default magnification for that type of text area.

Window User Data

You can store arbitrary data with a window using the `userdata` keyword with the **SetWindow** operation (page V-569). This is a topic of interest to advanced Igor programmers.

Each window has a primary, unnamed user data that is used by default. You can also store an unlimited number of different user data strings by specifying a name for each different user data string. The name can be anything you desire as long as it is a legal Igor name.

You can retrieve information from the default user data with the **GetWindow** operation (page V-225). To retrieve any named user data, you must use the **GetUserData** operation (page V-224).

Following is a simple example of user data using the top window:

```
SetWindow kwTopWin,userdata= "window data"  
Print GetUserData("", "", "")
```

Although there is no size limit to how much user data you can store, it does have to be generated as part of the recreation macro for the window when experiments are saved. Consequently, huge user data can slow down experiment saving and loading.

User data is intended to replace or reduce the usage of global variables. Named user data is intended for authors of packages that add features to existing windows. The primary author of a window should use the default, unnamed user data.

Chapters About Specific Windows

Detailed information about each type of window can be found in these chapters:

Window Type	Chapter
Command window	Chapter II-2, The Command Window Chapter IV-1, Working with Commands
Procedure windows	Chapter III-13, Procedure Windows
Help Browser Help windows	Chapter II-1, Getting Help
Graphs	Chapter II-12, Graphs
Tables	Chapter II-11, Tables
Layouts	Chapter II-16, Page Layouts
Notebooks	Chapter III-1, Notebooks
Control panels	Chapter III-14, Controls and Control Panels

Window Shortcuts

Action	Shortcut (<i>Macintosh</i>)	Shortcut (<i>Windows</i>)
To close a window:	Click the close button or press Command-W.	Click the close button or press Ctrl+W.
To kill a window with no dialog:	Press Option and click the close button or press Command-Option-W.	Press Alt and click the close button or press Ctrl+Alt+W.
To hide a window:	Press Shift and click the close button or press Command-Shift-W.	Press Shift and click the close button or press Ctrl+Shift+W.
To invoke the Window Control dialog:	Press Command-Y.	Press Ctrl+Y.
To send the top window behind all others:	Press Control-Command-E.	Press Ctrl+E.
To bring the bottom window on top of all others:	Press Shift-Control-Command-E.	Press Ctrl+Shift+E.
To send all windows that are completely visible behind all others:	Press Command-Option-E.	Press Ctrl+Alt+E.
To activate a recently activated window:	Press Command, click the main menu bar, and select from the Recent menu.	Press Ctrl, click the main menu bar, and select from the Recent menu.
To move a window to its preferred size and position:	Click the zoom button.	Press Alt and click the maximize button.
To move a window to its full-size position:	Press Shift-Option and click the zoom button.	Press Shift +Alt and click the maximize button.
To activate the command window:	Press Command-J.	Press Ctrl+J.
To clear the command buffer:	Press Command-K.	Press Ctrl+K.
To open the built-in procedure window:	Press Command-M.	Press Ctrl+M.
To cycle through all procedure windows:	Press Command-Option-M.	Press Ctrl+Alt+M.
To open the Help Browser:	Press Help or Command-?.	Press F1.
To find a phrase in a text window:	Press Command-F.	Press Ctrl+F.
To find the same phrase again:	Press Command-G. Press Command-Shift-G to search backward.	Press Ctrl+G. Press Ctrl+Shift+G to search backward.
To find the selected phrase:	Press Command-Control-H. Press Command-Shift-Control-H to search backward. This key combination can be changed through the Miscellaneous Settings dialog.	Press Ctrl+H. Press Ctrl+Shift+H to search backward.

