C4. ViewIt Window Commands

ViewIt supports a complete set of commands for opening, managing, and closing modal, modeless, and floating windows.

Name Number Parameters & Variables used

NewWnd -1201 a,b,c,d,wWindow

Opens a new ViewIt window according to parameters passed:

a = FWND resource ID (use -ID to keep window hidden)
b = initial window type (see "Windows" in ViewIt guide)
0 = MODAL
1 = MODELESS
2 = MODELESS FLOATING
3 = MODAL ALERT
c = 0 or byte size of linked record (see "Data Links" topic)
d = 0 or address of linked record (see "Data Links" topic)

and where wWindow returns with window's WindowPtr (or zero if window was not opened). If the FWND is not found, then ViewIt displays an alert and gives you the option of adding the FWND to the current res file at that time.

Alternatively, NewWnd can be used to open an FWND into an existing window (i.e., to replace its current contents):

a = FWND resource ID (use -ID to keep views hidden)
b = window whose contents are to be replaced
128 to 32767 = FWND ID of an existing window
other = WindowPtr of an existing window
c = 0 or byte size of linked record
d = 0 or address of linked record

where, again, wWindow returns the window's WindowPtr (or zero if an error occurs). Passing a negative FWND ID results in keeping all views hidden, which is useful in cases where the program needs to finish initializing the window's contents before making them visible. Once initialized, the hidden views can be shown with ShoCtl.

MODAL NOTES:

- If successive modal windows are opened, then their event loops must be "nested" so that each new modal window's NewWnd, MdlWnd, and EndWnd commands are contained within the previous modal window's event loop.
- Windows originally opened as modal cannot be later made modeless or floating with the MdlWnd command.
- Modal ViewIt windows can be shown or hidden at any time using ShoWnd and HidWnd. This will not affect the ordering of existing modal ViewIt windows. You might use HidWnd, for example, to hide a modal window that the user is finished with but that you need to keep around for a time before calling EndWnd.
- Any attempt to close with EndWnd (or switch to modeless state with MdlWnd) a modal window that is not the top modal window will result in closing (or switching to modeless) all windows above it. This provides a quick way to close multiple modal windows.
- Returning to your main event loop (i.e., where DoLoop is called) when modal windows are still open will cause FaceIt to close (or switch back to modeless) all modal windows.
- The toolbox call "FrontWindow" will not return the top modal window's window pointer, and fRec's fActiveWnd variable is set to nil when a ViewIt modal window is open. Use ViewIt's GetWnd command to get information about modal and modeless ViewIt windows. The window pointer of a new ViewIt window can also be obtained from wWindow after a successful call to NewWnd.

DRAWING NOTES: If calling NewWnd to open a new modeless window, note that FaceIt's fActive... variables and the current port are not updated until the next call to DoLoop. If you need to draw into such windows, or use ViewIt commands that are affected by the active window identity (such as when a = 0 is passed to GetCtl) before calling DoLoop, then add a call to DoUpdt (with d = 8 or -1) to force FaceIt to update its fActive... variables and reset the current port to the active window. Alternatively, you could use the window pointer returned by NewWnd in wWindow to reset the port, or pass the window pointer or FWND ID to identify a window when executing ViewIt commands like GetCtl.

Similar considerations apply when using NewWnd to open a new modal window. If you must draw into such a window before calling MdlWnd to handle events, then call MdlWnd with b = -2 once to get ViewIt to update the window and reset the current port before doing any drawing.

If you need to draw into a window that is not the active or front modal window, then call SetPort to first reset the current port to that window. This should also be done before using calls such as GlobalToLocal and LocalToGlobal

since these are a function of the current port. (Exception: If drawing from within a control driver or override procedure, then you don't have to worry about this since the current port is always reset to the port containing the control before a driver is called.)

COLOR NOTE: In some cases you may need to know whether an attempt to open a color window was successful or not (i.e., whether the program is being run on a Mac with Color QuickDraw). You can either check for the presence of Color QuickDraw directly (see fEnvFlags notes in "fRec Record" topic), or can examine the "rowBytes" field of the resulting window record (the 2-byte integer found at an offset of 6 bytes into the GrafPort or CGrafPort record). If rowBytes is negative, then the window is a color window.

REFCON USE: ViewIt does not make use of either the "windowKind" or "refCon" fields of a window's WindowRecord, so advanced programmers can make use of these to store program information.

EndWnd -1202 a,wiHit,wvHit,wcHit,wEvent

Disposes of the ViewIt window defined by parameter a, where a can be either an FWND ID, a window pointer, zero to close the top modal or active modeless window, or -1 to close all open modal ViewIt windows.

If the ViewIt window being closed is a modal window that lies below other modal windows, then all windows above it will also be closed (or made modeless again if they were modeless windows made modal by MdlWnd).

MODAL NOTE: If the window being closed is modal, then the variables wiHit, wvHit, wcHit, wClick, and wEvent are also restored to the values that they held when the window was opened with NewWnd (or made modal with MdlWnd). This feature facilitates the nesting of modal event loops designed around use of the "Hit" variables, but could be confusing in other cases if you forget that EndWnd changes these variables.

CURSOR NOTE: If the window being closed is visible, then ViewIt resets the cursor to an arrow before closing the window. If you prefer to have a different cursor shown while closing the window, then use HidWnd to hide the window before changing the cursor and calling EndWnd to dispose of the window.

MdlWnd -1203 a,b,c

Passes control to ViewIt to perform event handling for, or modify the current mode of, the window designated by a, where a can be either the associated FWND ID, the window's window pointer, or zero to designate the top modal or active modeless window. Parameter b designates the operation:

```
 2 = convert modal window back to floating modeless
 1 = convert modal window back to modeless
 0 = pass control to ViewIt for modal event handling
-1 = pass control but return w/o processing events
-2 = pass control and return after processing events
```

The latter modal operations (b = 0, -1, or -2) can be applied to both modal and modeless windows. The window is always brought to the front and made visible, and modeless windows are made modal until MdlWnd is called again with b = 1 or 2. MdlWnd is usually found between calls to NewWnd and EndWnd. See "Windows" topic in ViewIt guide for more info.

In rare cases, you may wish to process mouse down events within windows that are below a modal window. If c = 1 is passed to MdlWnd (when b = 0 or -2), then ViewIt will return such mouse events as raw events (instead of beeping): uMenuID = 0, wEvent.what = 1 (mouse down), and the window's window pointer is placed in wEvent.message. Your program should respond by either processing the event or by beeping if the event is ignored (to mimic the default behavior of modal windows).

NOTE: If MdlWnd is used to convert a modal window back to its previous modeless state (b = 1 or 2), then the variables wiHit, wvHit, wcHit, wClick, & wEvent are also restored to the values that they held when the window was made modal. As with EndWnd, this facilitates the nesting of modal event loops designed around use of the "Hit" variables, but could be confusing in other cases if you forget that MdlWnd changes these variables.

SizWnd -1204 a,b,c,d

Resizes the ViewIt window designated by a, where a can be either the associated FWND ID, window's window pointer, or zero to designate the top modal or active modeless window.

To resize the window, pass the window's new horizontal and vertical dimensions in parameters b and c, respectively. The size of the window may be limited by controls or views that are attached to the bottom or right sides of the window.

To "zoom" the window in or out (i.e., to mimic a hit in the window's zoom box), pass -1 in b and one of the following values in c: 7 = zoom in, 8 = zoom out, or 0 = zoom according to current state of window

(equivalent to hit in zoom box). The window must have a zoom box for this option to work.

To force ViewIt to just update the position and size of any attached controls without resizing the window, pass b = c = 0 to SizWnd. This is useful in cases where your program directly adds, removes, moves, or resizes attached controls or views using ViewIt or toolbox commands. Passing b = c = 0 also results in updating the standard zoom state of the window, which may be useful if you have used the toolbox call "MoveWindow" (vs. MovWnd) to move the window.

For modeless windows, parameter d can be optionally used to designate the amount of updating to be done at the time SizWnd is called, where its meaning is the same as that defined for the DoUpdt command.

NOTE: Do not attempt to use the toolbox call SizeWindow in place of SizWnd since the former will not readjust the size of controls that are attached to the window's sides.

MovWnd -1205 a,b,c,d

Moves the ViewIt window designated by a, where a can be either the associated FWND ID, window's window pointer, or zero to designate top modal or active modeless window. Parameters b, c, and d are the same as those supported by the utility command MovRec.

ShoWnd -1206 a,d

HidWnd -1207 a,d

Hides (HidWnd) or shows & selects (ShoWnd) the ViewIt window designated by a, where a can be either the FWND ID, the window's window pointer, or zero to designate top modal or active modeless window. The content of windows will be updated on the next call to DoLoop (if in modeless loop) or MdlWnd (if in modal loop).

In some cases you may need to have the content of the window updated before calling DoLoop or MdlWnd. For modeless windows, parameter d can be optionally used to designate the amount of updating to be done at the time HidWnd or ShoWnd is called, where its meaning is the same as that defined for the DoUpdt command. For modal windows, make a separate call to MdlWnd (with b = -2) or DoUpdt to force the window contents to be updated without losing control to ViewIt.

NOTE: Do not attempt to use the toolbox calls HideWindow, ShowWindow, or SelectWindow in place of Hid/ShoWnd since the former will not maintain proper window layering.

SavWnd -1224 a

Saves the FWND resource associated with the ViewIt window designated by a, where a can be either the FWND ID, the window's window pointer, or zero to designate top modal or active modeless window. If the window is opened at a fixed position ("Global Coordinates" option is checked in Window dialog), then SavWnd also saves the current position of the window as part of the FWND resource. SavWnd is often used to give users the option of saving window positions as part of saving other program-specific setup info.

NOTE: SavWnd does not call GetWVC to update the FWND in memory before saving it to disk, meaning that changes which your program directly makes to the window contents (via ShoCtl, ActCtl, AddCtl, etc.) will not be saved by SavWnd unless GetWVC is first called.

HlpWnd -1229 b,c,d

Opens ViewIt's main help window (if it is not already open) as a modeless window in a FaceIt or FaceSt-based program. If FaceIt or FaceSt is not in use, or ViewIt's on-line help resources are not available, then the window is not opened. Parameters b, c, and d are the same as those supported by the MovRec utility command for positioning the window, but are overridden by any settings saved in STR# 1211 via "Save Help Settings" in ViewIt's Help window.

NOTE: If using FaceIt, then FaceIt will automatically attempt to open the help window when DoInit is called. You can disable this feature by adding 64 to parameter c when calling DoInit.

DoUpdt -53 a,d

Updates window stuff according to the value of parameter d:

0 = no updating

1 = redraw all window contents that need updating

8 = update standard menu items and, if FaceIt/St is in use, update identity of active window and reset

current port to the active window
-1 = update all of above (all bits set)
When using FaceSt, for example, DoUpdt is called with d = 8 to update the active window-related variables when the active window is changed. A more common use is to call DoUpdt with d = 1 to redraw the content of an erased window before new drawing is done by the program. When using d = 1, parameter a can be optionally used to pass the WindowPtr of a specific window to be updated, otherwise all windows needing updating are redrawn.