Windows - Managing and Moving To Other Windows

| | Wind | ows · | - Managin | g and I | Mov | ing To Ot | her Wind | ows | |
|---|--|--|--|--|--|--|------------------------|------------------------|----|
| Operation | Keystroke | 1 | Function | 1 | | | No | ote . | |
| Window Operations See also: • ∑ Customize • ∑ Key-Chords • ∑ Frames • ∑ Speedbar | Emacs basic window management commands are bound to C-x o, C-x 0, C-x 1, C-x 2 and C-x 3 with some derivatives and support for multiple frames. These basic facilities can be extended by several built-in and external packages: • windmove, built-in, activated by PEL, but not using the standard wind move key bindings (which use Shift with cursors) to preserve ability to shift-mark when moving across text with cursor. • winner, also built-in, which provides the ability to restore previous window pane layouts. • PEL activates it when pel-use-winner user option is t. • ace-window, an external package, which extends the C-x o command by displaying Ace target in the windows' upper left corner to quickly select the target window to move to and possibly operate on. • key-chord, an external package, that PEL use (when pel-use-key-chord user option is t) to activate dual-key chords to move across windows. • Windows can be dedicated to specific buffers, for example by Speedbar (see Speedbar). • Several windows with the same buffers can operate as a single flow with follow mode. PEL provides extra commands and key bindings: | | | | | | | | |
| | It adds several key bindings under the <f11> key prefix. These are available in both graphics and terminal modes.</f11> It adds several key bindings under the <f11> key prefix. These are available in both graphics and terminal modes.</f11> On macOS, in graphics mode only, the % key is mapped to the super prefix key (s-). On Windows, the Menu key is mapped to the hyper key. Below the \$ icon is used to represent the Menu key under Windows. In graphics mode, mouse operations are available. They can also be enabled in terminal mode, with the xterm-mouse-mode enabled. With PEL, use <f11><f12> to toggle the xterm-mouse-mode operations on windows can be applied to windows in other frames, whether Emacs is running in graphics mode or in terminal mode.</f12></f11> In terminal mode only one frame is visible at a time though. | | | | | | | | |
| Open this PDF file. See also: <u>Nelp/Info</u> | <f11> w <f1> (pel-help-pdf &optional OPEN-WEB-PAGE)</f1></f11> | | nal OPEN- | Open the local copy of the <u>Windows</u> PDF file unless a command prefix (like C-u) was used. In that case it opens the Github-hosted file instead. | | | | | |
| <u>∑ Customize</u> PEL window control | <f11> w <f2></f2></f11> | (pel-customize-pel &optional OTHER-WINDOW) | | koptional | Customize PEL Window support. • If OTHER-WINDOW is non-nil (use C-u), display in other window. | | | | |
| ∑ Customize Emacs window control | <f11> w <f3> (pel-customize-library &optional OTHER-WINDOW)</f3></f11> | | ry &optional | | omize Emacs Windov er, windmove. | v support groups: v | vindows, ace-window, a | ce-window-display, | |
| PEL Window Management Hydra | With PEL user option pel-use-hydra set to t, PEL activates the hydra external package and also creates a Hydra set of keys to help speed up navigation and management of windows. These keys are identified in the table below. To start this Hydra, hit the <f7> key, then hit one of the following keys once or several times. The keys that are in the PEL window hydra are all identified below with a <f7> prefix, but after typing <f7> once, you can hit several other window hydra without typing the <f7> prefix again. While active the Hydra Hint is normally shown in the minibuffer (like the one shown below). While the Hydra is active use the ? key to toggle it off or bar on. To have the Hydra hint off when the Hydra activates set the hydra-is-helpful user option to nil (but then you can still toggle it on/off with ?. To cancel the Hydra hit the <f7> key again or use a command not available in the windows management hydra. The name of the PEL window hydra commands are not listed below. They all have a name that begins with pel-∑wnd/ and ends with the same name at the command function listed in the Function column. For example, pel-∑wnd/windmove-up is bound to <f7> <up>. A snapshot of the window management hydra hint menu that shows up in the minibuffer area as soon as one of its keys is pressed is shown below. Use the q key to quit from buffers that can be dismissed like the "Help" buffer. It also changes the buffer visible in the normal windows. You can also use the total particular of the visible in the current window. Move Resize Split Split Layout Close/Buffer Close. C-upp: above n: next layout 0: this window C-S-<up>: hint C-S-<up>: above C-S-<up>: hint C-S-<up>: hint C-S-<up>: c-S-<up>: hint C</up></up></up></up></up></up></up></up></up></up></up></up></up></up></up></up></up></up></up></up></up></up></up></up></f7></f7></f7></f7></f7></f7> | | | | | al other window hydra to toggle it off or back ff with ?. In the same name as hown below. ws. You can also use | | | |
| | The ace-window | bound to | C-x o key provides a | a partially ove | rlappin | g feature set but they | differ in their key a | assignments. See belov | V. |
| window - C-u: swap - C-u C-u: delete • Move to other window • Move to specified window Ace target • Operate on specified window See also: © Customize | | | (ace-window ARG) (windmove-up & optional ARG) | | Select another window in cyclic ordering of windows. This is Emacs default behaviour for this key. And PEL's default: pel-use-ace-window = nil. Change it to activate the functionality described in next row. Move to (and possibly operate on) window selected by an Ace target code. With only 2 windows, move to the other window. With 3 windows or more: display an Ace target in the windows' upper left corner: it identifies the window target. Type this number to move to the window, or Type one of the following letters, followed by the target number to move to the target window and operate on it: x - delete window m - swap windows M - move window j - select buffer n - select the previous window u - select the previous window v - split window fairly, either vertically or horizontally v - split window vertically b - split window horizontally o - maximize current window ? - show these command bindings This supports selecting windows in other frames (both in graphics and terminal mode) In graphics mode the other Emacs frames are in other OS window. | | | | |
| Move to window above | | | | | In text terminal mode, other Emacs frames are hidden (as they occupy the exact same OS window): just one Emacs frame is displayed. ★ An argument can be used to perform more operations: To force a window number prompt, use any negative prefix (including just typing Calone). Useful with several frames when current frame has 1 or 2 windows active. Prefixed with one C-u, does a swap between the selected window and the current window, so that the selected buffer moves to current window (and current buffer moves to selected window). The PEL <f11> w x key does the same (but does not prompt when there are only 2 windows.)</f11> Prefixed with two C-u's, deletes the window identified by the window number. ★ Requires the ace-window external package. A PEL downloads, installs and activates it when the pel-use-ace-window user option is set to t. Demo: C'est la Z, video 5 Select the window above the current one. With no prefix argument, or with prefix argument equal to zero, "up" is relative to the position of point in the window; otherwise it is relative to the left edge (for positive ARC or the right edge (for negative ARG) of the current window. If no window is at the desired location, an error is signaled. ★ With PEL, the yu key-chord is also available when key-chord is available and active. See ★ Key-Chords. | | | | |

| Operation | <u>Keystroke</u> | Function | <u>Note</u> | | |
|--|--|--|---|--|--|
| Move to window below | • <f11> <down> • <esc> <down> • %-<down> • * -<down> • f7> <down> • in the state of the state of</down></down></down></down></esc></down></f11> | (windmove-down &optional ARG) | Select the window below the current one. • With no prefix argument, or with prefix argument equal to zero, "down" is relative to the position of point in the window; otherwise it is relative to the left edge (for positive ARG) or the right edge (for negative ARG) of the current window. • If no window is at the desired location, an error is signaled. With PEL, the bn key-chord is also available when key-chord is available and active. See See-Chords . | | |
| Move to window at right | • <f11> <right> • <esc> <right> • %-<right> • \$ -<right> • ⟨f7> <right> • ik</right></right></right></right></esc></right></f11> | (windmove-right &optional ARG) | Select the window to the right of the current one. • With no prefix argument, or with prefix argument equal to zero, "right" is relative to the position of point in the window; otherwise it is relative to the top edge (for positive ARG) or the bottom edge (for negative ARG) of the current window. • If no window is at the desired location, an error is signaled. □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ | | |
| Move to window at left | <pre> • <f11> <left> • <esc> <left> • %-<left> • \$ -<left> • <f7> <left> • fg</left></f7></left></left></left></esc></left></f11></pre> | (windmove-left &optional ARG) | Select the window to the left of the current one. • With no prefix argument, or with prefix argument equal to zero, "left" is relative to the position of point in the window; otherwise it is relative to the top edge (for positive ARG) or the bottom edge (for negative ARG) of the current window. • If no window is at the desired location, an error is signaled. With PEL, the fg key-chord is also available when key-chord is available and active. See ★ Key-Chords. | | |
| Move point to next window | <f11> w o</f11> | (pel-other-window) | Execute (other-window 1). • Useful when 'other-window' has been remapped to something like 'ace-window' and want to see where the <i>next</i> window is. | | |
| Move point to previous window | <f11> w 0</f11> | (pel-other-window-backward &optional N) | Select Nth previous window. • n defaults to 1: meaning direct previous window. • This is the inverse of what does the standard (other-window). • This command might be useful when ace-window is not used. | | |
| Exchange windows | • <f11> w x • <f7> x</f7></f11> | (ace-swap-windows) | Swap buffers of the current window with another. If 3 windows or more, a single digit shows up in the top-left corner identifying the number to type to swap to this window. Requires the <u>ace-window</u> external package. PEL downloads, install and activates it when the <u>pel-use-ace-window</u> user options is set to t. | | |
| Close/Create Windows | | used to create and remove windows or two sets of four PEL commands be | | | |
| Close this windows | • C-x 0 • <f7> 0 • <f7> d</f7></f7> | (delete-window &optional WINDOW) | This just closes the window and moves the cursor to the next window. | | |
| Kill current buffer and close window See also: <u>Seuffers</u> | • C-x 4 0 • <f7> k</f7> | (kill-buffer-and-window) | Kill the current buffer and delete the selected window. | | |
| Close a window identified by number | <f11> w k</f11> | (ace-delete-window) | Delete a window selected by a number, a number shown in the top-left corner of the window. Requires the <u>ace-window</u> external package. PEL downloads, installs and activates it when the <u>pel-use-ace-window</u> user options is set to t. | | |
| Close all other windows | • C-x 1 • <f7> 1 • <f7> .</f7></f7> | (delete-other-windows &optional WINDOW) | Make current window fill its frame. | | |
| Maximize one window, identified by number | <f11> w m</f11> | (ace-maximize-window) ——————————————————————————————————— | Maximize a window. Close all windows except the window selected by number, a number shown in the top-left corner of the window. Requires the <u>ace-window</u> external package. The old versions used ace-window-maximize, but newer versions use ace-delete-maximize-windows. PEL uses the one that is available. PEL downloads, install and activates it when the <u>pel-use-ace-window</u> user options is set to t. | | |
| Create new window below | • C-x 2 • <f7> 2 • <f7> -</f7></f7> | (split-window-below &optional SIZE) | Split the selected window into two windows, one above the other. • The selected window is above. The newly split-off window is below and displays the same buffer. • Note that Emacs default behaviour attempts to maximize the view into the current buffer when splitting the buffer into 2 windows. This means that the cursor will not be located in the same position in the new window. To change this behaviour and keep the same point in both windows, execute (setq split-window-keep-point nil). The PEL packages does that. | | |
| Create new window at right | • C-x 3 • <f7> 3 • <f7> </f7></f7> | (split-window-right &optional SIZE) | Split the selected window into two side-by-side windows. The selected window is on the left. The newly split-off window is on the right and displays the same buffer. | | |
| Create window at cursor direction | <pre> <f11> C-<right> <f11> C-<left> <f11> C-<down> <f11> C-<up> <f7> C-<right> <f7> C-<right> <f7> C-<left> <f7> C-<left> <f7> C-<left> <f7> C-<down> <f7> C-<down> <f7> C-<up> <f7> C-<up> <f7> C-<up> </up></f7></up></f7></up></f7></down></f7></down></f7></left></f7></left></f7></left></f7></right></f7></right></f7></up></f11></down></f11></left></f11></right></f11></pre> | (pel-create-window-right) (pel-create-window-left) (pel-create-window-down) (pel-create-window-up) | Create a window at the location pointed by the cursor's direction, and keep point inside the new window. These are 4 different commands, one for each of the available cursors: <right>, <left>, <down> and <up>. The <f11> commands are stand alone commands, while the <f7> are part of the Hydra for Windows Management.</f7></f11></up></down></left></right> | | |
| Close a window at cursor direction | <pre></pre> | pel-close-window-right) (pel-close-window-left) (pel-close-window-down) (pel-close-window-up) | Kill window pointed by the cursor's direction. ★These are 4 different commands, one for each of the available cursors: <right>, <left>, <down> and <up>. ★The <f11> commands are stand alone commands, while the <f7> are part of the PEL Hydra for Windows Management.</f7></f11></up></down></left></right> | | |
| Resize Window | The following commands are used to change the current window size. None of these commands are easy to re-type quickly. The best way to use them is to type them once and then use the repeat key C-x z once and then repeat more by only typing ' z '. The PEL package also binds the <f5></f5> key to repeat. | | | | |
| Grow window taller | • C-x ^ • <f11> w s V • <f7> x</f7></f11> | (enlarge-window DELTA &optional HORIZONTAL) | Grow window taller by DELTA lines (defaults to 1), specify more with C-u n (or M- n) argument prefix. | | |
| Shrink window smaller | • <f11> w s v • <f7> v</f7></f11> | (shrink-window DELTA &optional HORIZONTAL) | Shrink height of window by DELTA lines (defaults to 1), specify more with C-u n (or M- n) argument prefix. | | |
| Grow windows wider | • C-x } • <f11> w s H • <f7> H</f7></f11> | (enlarge-window-horizontally DELTA) | Enlarge the current window horizontally. | | |

| <u>Operation</u> | <u>Keystroke</u> | Function | <u>Note</u> | | | |
|---|---|---|--|--|--|--|
| Shrink window narrower | • C-x { • <f11> w s h • <f7> h</f7></f11> | | Reduce the width of the current window. | | | |
| Reduce current window size if buffer is smaller than window | • C-x - (shrink-window-if-larger-than buffer & optional WINDOW) | | Shrink height of current window if its buffer doesn't need so many lines. • More precisely, shrink window vertically to be as small as possible, while still showing the full contents of its buffer. • Do not shrink window to less than 'window-min-height' lines. Do nothing if the buffer contains more lines than the present window height, or if some of the window's contents are scrolled out of view, or if shrinking this window would also shrink another window, or if the window is the only window of its frame. | | | |
| Make all windows the same size | • C-x + • <f11> W S = (balance-windows & optional WINDOW-OR-FRAME)</f11> | | Balance the sizes of windows of WINDOW-OR-FRAME. WINDOW-OR-FRAME is optional and defaults to the selected frame. If WINDOW-OR-FRAME denotes a frame, balance the sizes of all windows of that frame. If WINDOW-OR-FRAME denotes a window, recursively balance the sizes of all child windows of that window. | | | |
| Quick Window Layout Change | The following commands flip | the layout of 2 windows: the current | and <i>next</i> window between 2 horizontal windows to 2 vertical windows and vice versa. | | | |
| Flip 2 horizontal windows to 2 vertical ones | • <f11> w v • <f7> M-v (pel-2-vertical-windows)</f7></f11> | | Convert 2 horizontal windows into 2 vertical windows. Flip the orientation of the current window and its next one. The next window is placed at the right of the current window. | | | |
| Flip 2 vertical windows to 2 horizontal ones | • <f11> w h • <f7> M-h</f7></f11> | (pel-2-horizontal-windows) | Convert 2 horizontal windows into 2 horizontal windows. • Flip the orientation of the current window and its next one. • The next window is placed below the current one. | | | |
| Window Layout History | The following commands allow you to restore a previously used window layout. They depend on the winner package, a package that is part of the standard Emacs. PEL activates them when pel-use-winner user option is t. | | | | | |
| Restore an earlier window configuration | • C-c <left> (winner-undo) • <f11> w p • <f7> p</f7></f11></left> | | Switch back to an earlier window configuration saved by Winner mode. In other words, "undo" changes in window configuration. | | | |
| Restore a more recent window configuration | • C-c <right> • <f11> w n • <f7> n</f7></f11></right> | (winner-redo) | Restore a more recent window configuration saved by Winner mode. | | | |
| Open Buffer in another window | and the other does not. Und | er PEL both commands are bound to | dide another window. One command select that other window (move point to that window) of the IDO version of the command when the pel-use-ido customization variable is set to t, g provides more information at the prompt. | | | |
| Select buffer in other window | • C-x 4 b • <f11> w B</f11> | (ido-switch-buffer-other-window) | Select buffer bufname in another window (switch-to-buffer-other-window). See <u>Select Buffer</u> . | | | |
| Display buffer in other window, don't select the other window. | • C-x 4 C-o • <f11> w b</f11> | (ido-display-buffer) (display-buffer BUFFER-OR-NAME & Optional ACTION FRAME) | Display a buffer in other window but don't select it. When pel-use-ido is customized to t, (ido-display-buffer) is used, which prompts and provides easy to select list of available buffer names. Otherwise the standard Emacs (display-buffer) is used prompting without showing the available buffers. | | | |
| Dedicated Windows | Emacs windows can be dedicated to specific buffers in such a way that future windows operations do not affect the dedicated windows. The following commands help you manage dedicated windows. | | | | | |
| Show dedicated status of current window | <f11> w d ?</f11> | (pel-show-window-dedicated- status) | Display the dedicated status of the current window in the echo area (the minibuffer). | | | |
| Toggle dedicated status of current window | <f11> w d d</f11> | (pel-toggle-window-dedicated) | Toggle the dedicated status of the current window. Use with care. | | | |
| Follow Mode | implement extra code as sug | ws mode which applies all scroll com gested by the Emacs Wiki Scroll All de using 3 windows | | | | |
| See also: Scrolling | Text in the first window goes to the bottom and then | window it continues there. | When Emacs follow-mode is used on 2 or more windows, these windows show the text of the same buffer spread across these windows that act as a one continuous stream. Follow mode is a minor mode that combines windows into one tall virtual window. This is accomplished by two main techniques: The windows always displays adjacent sections of the buffer. This means that whenever one window is moved, all the others will follow. (Hence the name Follow mode.) Should point (cursor) end up outside a window, another window displaying that point is selected, if possible. This makes it possible to walk between windows using normal cursor movement commands. Follow mode comes to its prime when used on a large screen and two or more side-byside windows are used. The user can, with the help of Follow mode, use these full-height windows as though they were one. | | | |
| Toggle follow-mode See also: Scrolling | • <f11> w f • <f11> f</f11></f11> | (follow-mode &optional ARG) | Toggle Follow mode. With a prefix argument ARG, enable Follow mode if ARG is positive, and disable it otherwise. | | | |
| Scrolling Window | | o scroll the window text, see the S | crolling page. | | | |

Windows - Reference

| Topic/URL | Comment |
|---|--|
| GNU Emacs — Displaying a Buffer in a Window | Describes the Emacs features related to displaying buffers inside windows. |
| GNU Emacs Lisp — Displaying Buffers — The Zen of Buffer Display | Describes the rules Emacs tries to use to control the creation of new windows when they are created dynamically from commands. |
| | |