↖ Topic Index

# Frames

| Operation | Keystroke | Function | Note |
|---|---|---|---|
| **Emacs Frames**<br>See **Quick intro**<br>**Emacs Wiki Glossary** | • Emacs calls frames what modern graphical Operating Systems call “windows”, or more specifically “OS windows”.<br>• Emacs supports multiple frames, both when Emacs works in graphical mode and in text terminal mode.<br>  • In terminal mode, only one frame is visible at any given time, the other frames are hidden. The current frame number is shown on the mode line.<br>  • Using multiple frames is a powerful way to keep track of development contexts, with each frame having its own set of windows. | | |
| **Open this PDF file.**<br>See also: **Help/Info** | **<f11> F <f1>** | (**pel-help-pdf** &optional OPEN-WEB-PAGE) | Open the **Frames** local PDF. If the prefix argument (like **C-u** or **M--**) is used, then it opens the remote GitHub hosted raw PDF instead. If the **pel-flip-help-pdf-arg** user-option is set it’s the other way around. |
| **Customize PEL frame control** | **<f11> F <f2>** | (**pel-customize-pel** &optional OTHER-WINDOW) | Customize PEL frame management support.<br>• If OTHER-WINDOW is non-nil (use **C-u**), display in other window. |
| **Customize Emacs frame control** | **<f11> F <f3>** | (**pel-customize-library** &optional OTHER-WINDOW) | Customize Emacs frame management support. |
| | | | |
| **Enter/Exit full screen** | **<f11> <f11>** | (**pel-toggle-frame-fullscreen**) | Toggle frame fullscreen mode on/off in graphics mode. |
| | • In Terminal mode, issue error to show how to use the OS keystrokes to toggle the fullscreen mode. For example it will tell you to use **⌘-C-f** when Emacs is running inside a Terminal.app frame. 🚧 Unfortunately the information is only provided for macOS Terminal.app and iTerm.app.<br>✂ Standard GNU Emacs normally binds **<f11>** to (toggle-frame-fullscreen). But PEL use **<f11>** as a prefix key, therefore it rebinds it. | | |
| **Set Frame Font** | With Emacs running in graphics mode, you can change the font of all windows with the menu-set-font command. | | |
| **Change font of current Frame**<br>See also: **Faces/Fonts** | **<f11> F F** | (**menu-set-font**) | Interactively select a font and make it the default on all frames.<br>• Set the default on both the existing and future frames in the current session. It is not persistent, so next time you start Emacs the default font is used. |
| **Move to another graphical frame using cursors** | The following commands move point to another graphical Emacs frame existing in the direction selected by the cursor key.<br>📦 **framemove** 🔄 PEL installs and activates this when the **pel-use-framemove** user option is set to **t**. Also available in the Emacs Wiki framemove .<br>⚠ These four commands commands only work in **graphics mode.** | | |
| **Move to graphical frame above** | • **<f11> S-<up>**<br>• **<Esc> S-<up>** | (**fm-up-frame**) | Move point to frame above current frame (if one exists and is located there). |
| **Move to graphical frame below** | • **<f11> S-<down>**<br>• **<Esc> S-<down>** | (**fm-down-frame**) | Move point to frame below current frame (if one exists and is located there). |
| **Move to graphical frame at right** | • **<f11> S-<right>**<br>• **<Esc> S-<right>** | (**fm-right-frame**) | Move point to frame at right of the current frame (if one exists and is located there). |
| **Move to graphical frame at left** | • **<f11> S-<left>**<br>• **<Esc> S-<left>** | (**fm-left-frame**) | Move point to frame at left of the current frame (if one exists and is located there). |
| **Manage Frames** | Emacs frame operations work in **both** modes: graphics and text terminal. In graphics mode a new frame is creating a separate OS frame.<br>• In text terminal mode a new frame is inside the **same** OS frame: it simply hides the other(s) existing frame(s). The mode line identifies the currently active frame number as “F1”, “F2”, etc… The windows numbers are all part of the space number-space. | | |
| **Show frame count** | **<f11> F ?** | (**pel-show-frame-count**) | Display the number of Emacs active frames in the mini-buffer. |
| **Delete this frame**<br>See **Windows Hydra** | • **C-x 5 0**<br>• **<f11> F 0**<br>• **⌘-w**<br>✱ **<f7> M-F** | (**delete-frame** &optional FRAME FORCE) | Delete FRAME, permanently eliminating it from use.<br>• FRAME must be a live frame and defaults to the selected one. |
| **Delete all other frames** | • **C-x 5 1**<br>• **<f11> F 1** | (**delete-other-frames** &optional FRAME) | Delete all frames on FRAME’s terminal, except FRAME. |
| **New Frame below**<br>See **Windows Hydra** | • **C-x 5 2**<br>• **<f11> F 2**<br>• **⌘-n**<br>✱ **<f7> M-f** | • (**make-frame-command**)<br>• (**make-frame** &optional PARAMETERS) | Make a new frame, on the same terminal as the selected frame.<br> On macOS in graphics mode only: make-frame is called for **⌘-n**, it has the same effect as make-frame-command. |
| **Display buffer in other (next) frame** | • **C-x 5 C-o**<br>• **<f11> F b** | (**display-buffer-other-frame** BUFFER) | Display a buffer preferably in another frame. |
| **Tear window in frame**<br>See **Windows Hydra** | **C-x w ^ f**<br>• **<f11> w i f**<br>✱ **<f7> F** | (**tear-off-window** CLICK) | Delete the selected window, and create a new frame displaying its buffer.<br>• See: **Windows** |
| **Select another frame** | • In text terminal mode: iterate through all frames, make the next frame visible in the terminal window and update the frame number shown on the mode line.<br>• In graphics mode: with no/nil argument (the default): iterate through visible frames, with ARG non nil: iterate through all frames (included iconified frames). | | |
| **Select next frame**<br>• the frame with the next higher frame number | • **<f11> F n**<br>• **⌘-`**<br>✱ **<f7> }** | (**pel-next-frame** ARG) | Make next frame visible.<br> The **⌘-`** key is a macOs command that moves to the next frame of the same application: no Emacs code is invoked but the effect is the same in graphics mode. |
| **Select previous frame**<br>• the frame with the next lower frame number | • **<f11> F p**<br>• **⌘-~**<br>✱ **<f7> {** | • (**pel-previous-frame** ARG)<br>• (**ns-prev-frame**) | Make previous frame visible.<br> The **⌘-`** key is a macOs command that moves to the next frame of the same application: no Emacs code is invoked but the effect is the same in graphics mode. |
| **Move cursor to other (next) frame** | • **C-x 5 o**<br>• **<f11> F o** | (**other-frame** ARG) | Activate the other frame. Numeric argument identifies frame in Z order. |
| **Open in frame** | The following commands open a buffer, file or **Dired** buffer in a frame. | | |
| **Select buffer in other frame** | • **C-x 5 O**<br>• **<f11> F O** | (**switch-to-buffer-other-frame** BUFFER-OR-NAME &optional NORECORD) | Prompt for buffer and open in other frame. |
| **Find file in other (next) frame in read-only** | • **C-x 5 r**<br>• **<f11> F r** | (**find-file-read-only-other-frame** FILENAME &optional WILDCARDS) | Edit file read-only in other frame with name obtained via minibuffer. |
| **Find file in other (next) frame** | • **C-x 5 f**<br>• **C-x 5 C-f**<br>• **<f11> F f** | (**find-file-other-frame** FILENAME &optional WILDCARDS) | Switch to/open another file and show it in another frame. |
| **Run Dired in other (next) frame** | • **C-x 5 d**<br>• **<f11> F d** | (**dired-other-frame** DIRNAME &optional SWITCHES) | "Edit" a directory. Like ‘dired’ but makes a new frame.<br>• See **Dired** |
| **macOS frame** | The following commands are available under macOS for Emacs running in graphical mode only. | | |
| **Hide Emacs frame** | **⌘-h** | (**ns-do-hide-emacs**) | On macOS in graphics mode only: hide Emacs frame.<br>• Retrieve it via the **⌘-<tab>** key. |
| **Hide all other applications** | **⌘-H** | (**ns-do-hide-others**) | On macOS in graphics mode only: hide all other applications, except Emacs.<br>• Retrieve them via the **⌘-<tab>** key. |
| **Iconify frame** | **⌘-m** | (**iconify-frame** &optional FRAME) | On macOS in graphics mode only: iconify the frame. |