Key-Chords

Action	Keystroke	Function	Note
Two Character Key-chords	With PEL user option **pel-use-key-chord** set to **t** or to **use-from-start**, • PEL activates the **key-chord** external package • PEL activates the global and mode-specific key chord bindings identified in the **pel-key-chords** user option. • If **pel-use-key-chord** is set to **use-from-start** it activates the key-chords when Emacs starts, otherwise you must first activate the key-chord mode with the key-chord-mode command which PEL maps to the keystroke **<f11> M-K**. PEL provides a set of pre-defined key-chords in the **pel-key-chords** user option and maps the the **<f11> <f1> K** to quickly access the PEL key-code customize buffer. and edit these values. You can add, delete or edit any of the provided key-chords, which provide examples of the ways to define your own key-chords. The list of key-chords PEL pre-defines and provides as default are show in the rows below. • A key chord is a group of 2 normal, non-modifier keys that must be typed simultaneously to activate the action identified in the key chord definition. • Here, we are not talking of something like the normal Emacs key bindings like C-s, where the Control key and the s key are type together to do a CONTROL-S or where M-b represents using the Meta key and the b key together. The key-chords discussed here allow you to define actions when you type, for example, the key 'j' and the key 'k' together, or when you type the '.' key twice quickly. *When the key-chord-mode is active* these special key-chord events are triggering the action you key-chord definition identifies. If the key-chord-mode is off, you get the normal Emacs behaviour of inserting the two keys inside the current buffer at point location.		
Toggle key-chord mode	**<f11> M-K**	(**key-chord-mode** ARG)	Toggle key chord mode. • With positive ARG enable the mode. With zero or negative arg disable the mode. • A key chord is two keys that are pressed simultaneously, or one key quickly pressed twice. Requires the key-chord external package. PEL activates it when **pel-use-key-chord** user option is **t**.
PEL Key-chords	The following rows describe the key-chords PEL defines by default in the **pel-key-chords** user option. • You can use them when the key-chord-mode is active. • You can also decide to change them if they do not suit you, delete or add new ones by customizing the pel-key-chords user option. PEL provides a key binding to quickly access the customize buffer for key-chord control: **<f11> <f1> K** • The **pel-key-chords** user option has complete docstring that describes how to add news values.		
Key Chords (See also:∑ Customize	**<f11> <f1> K**	(**pel-customize-key-chords** &optional OTHER-WINDOW)	Customize PEL Key Chord support. • If OTHER-WINDOW is non-nil (use **C-u**), display in another window.

PEL Pre-defined key-chords	• PEL default for **pel-key-chords** are identified in the tables of this document with the characters underlined.
	  • In some cases the key-chord is a simple binding to execute a command or an Emacs Lisp lambda form. In that case the 2 key-chord keys are shown in the keystroke column alone, simply underlined.
	  • In other cases, the key-chord inserts characters and execute commands. In such as case, the 2 key-chord keys are also shown in the keystroke column alone, but instead of describing the function in the function column, the cell shows the key-chord string which represent both the character inserted and the key code for the command.
	    • For example, the key-chord that consist of typing the **<** key and the **>** key together is represented as the **<>** key-chord and the expansion is show as **"<>\C-b"** . The effect is to insert both angle brackets and put point in between, since **C-b** is bound to to command backward-char.
	• The **color of the key-chord** corresponds to the availability of the commands used, if any. A key-chord that depends only on Emacs standard commands or simple characters is therefore shown in black.		

Insert <> and place point between them	**<>**	**<>\C-b**	Global: available in all modes.
Insert [ ] and place point between them	**[]**	**[]\C-b**	Global: available in all modes.
Insert {} and place cursor between	**{}**	**{\n\n}\C-p\C-p**	Available in c-mode and c++-mode
Move to window above	**yu**	(**windmove-up** &optional ARG)	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 ARG) or the right edge (for negative ARG) of the current window. • If no window is at the desired location, an error is signaled. Global: available in all modes.
Move to window below	**bn**	(**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. Global: available in all modes.
Move to window at left	**fg**	(**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. Global: available in all modes.
Move to window at right	**jk**	(**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. Global: available in all modes.
Indent rigidly	**<tab>q**	(**pel-indent-rigidly &optional N**)	Indent rigidly the marked region or current line N times. • **If a region is marked**, it uses 'indent-rigidly' and provides the same prompts to control indentation changes. • **If no region is marked**, it operates on current line(s) identified by the numeric argument N (or if not specified N=1): • N = [-1, 0, 1] : operate on current line • N > 1 : operate on the current line and N-1 lines below. • N < -1 : operate on the current line and (abs N) -1 lines above. Indent all lines starting in the region. • If called interactively with no prefix argument, activate a transient mode in which the indentation can be adjusted interactively by typing **<left>**, **<right>**, **<S-left>**, or **<S-right>**. — These commands activate a transient mode where Emacs prompts for extra keys to control how to indent. Indenting and un-indenting is possible. The capabilities are controlled by the variable *indent-rigidly-map* with by default provides: • **S-<right>** indent-rigidly-right-to-tab-stop • **S-<left>** indent-rigidly-left-to-tab-stop • **<right>** indent-rigidly-right • **<left>** indent-rigidly-left Typing any other key deactivates the transient mode.
Correct mode at point	**4r**	(**flyspell-correct-word-before-point** &optional EVENT OPOINT)	Pop up a menu of possible corrections for misspelled word before point. Available when current buffer has flyspell-mode or flyspell-prog-mode enabled.

Action	Keystroke	Function	Note
Find file at point	`6y`	**(pel-find-file-at-point-in-window** &optional N)	Open the file, library or the URL, named at point. • If the string identifies a URL, the function opens the page in the default browser. • If the string identifies a file name, the file is opened in Emacs in the window identified by the N argument. 8: up, 2: down, 4:left, 5:current, 6:right, 0: other, negative: new. Selecting Minibuffer, inexistent or dedicated window is not allowed. • If the file is not found, the function prompts. If the name corresponds to an Emacs library file, you can type **`l`** to open the library. You can also edit the file name collected before attempting to open it again. Or quit. • If the file name is followed by line and column numbers the point is moved to that position. More information available in the command's help docstring. Global: available in all modes.

Search word at point from top of current buffer	`.;`	**(pel-search-word-from-top** &optional N)	Search word at point from top/bottom of buffer in window identified by N.
			• **Search direction**:
			  • If N is nil, 0 or larger, perform a search-forward from the top of the buffer in window identified by N.
			  • If N is negative: perform a isearch-backward from the bottom of the buffer in the window selected by the absolute value of N.
			• **Window selection**:
			  • If N is not specified, nil, 1, 3, 7 or 9 and larger: search in current window.
			  • If N is 0: : search in other window
			  • If N in [2,8] range, search in window identified by the direction corresponding to the cursor in a numeric keypad:
			  8 := 'up
			  4 := 'left 5 := 'current 6 := 'right
			  2 := 'down
			• **Temporary word mode toggle**: detecting a '*word*' is affected by the subword-mode and superword-mode. When searching in current buffer, the following values of N temporary toggle the mode when grabbing the word:
			  • If N is 7: temporary toggle **subword-mode** to grab the word.
			  • If N is 9: temporary toggle **superword-mode** to grab the word.
			
			• Explicitly selecting the minibuffer window, or a non-existing window is not allowed, and search is done in current window.
			• Searched word is remembered and can be used again to repeat an interactive search with **`C-s`** or **`C-r`**.
			• Position before searched word is pushed on the mark ring.
			☝Using **superword-mode** allows you to search for function names in buffer for programming languages. If you do not want to change the mode but want to search for the word as interpreted by the other state of the mode type the command with N equal to 9: **`M-9 <f11> s .`**
			Global: available in all modes.

Key-Chords — References 🚧

Topic & Link	Description
key-chord @ MELPA	This page shows the doc coming from the key-chord.el. It's where PEL gets the file from.
Key Chords @ Emacs Wiki	Some interesting discussion about key-chord.
key-chord.el @ Emacs Wiki	