
























Cursor / Multiple-Cursors

Operation	Keystroke	Function	Note
Controlling Emacs' Cursor See also: 🔗 Customize	You can control Emacs cursor color and shape when Emacs is running in graphical mode (X mode). When Emacs is running internal mode, there are no standard way to manipulate the cursor for all terminals and in some the controls are very limited.		
	  When Emacs runs in graphics mode, PEL provides the following user options cursor control for Emacs running in graphics mode: <ul style="list-style-type: none">• pel-cursor-overwrite-mode-color : Selects the cursor color when overwrite-mode is active. Default is black on white background and white on black background.• pel-cursort-type-when-mark : Selects the cursor type (shape) when mark is active. Default to no cursor type change. Set it to a different type than 'cursor'. A popular setting is to use 'bar' type when mark is on to help see what is in the region.		
Open this PDF file. See also: 🔗 Help/Info	<f11> m <f1>	(pel-help-pdf &optional OPEN-WEB-PAGE)	Open Open the local copy of the 🔗 Cursor PDF file unless a command prefix (like C-u) was used. In that case it opens the Github-hosted file instead.local PDF file.
Customize PEL Cursor control	<f11> m <f2>	(pel-customize-pel &optional OTHER-WINDOW)	Customize PEL support for cursor and multiple-cursors. <ul style="list-style-type: none">• If OTHER-WINDOW is non-nil (use C-u) , display in other window.
Customize Emacs cursor control.	<f11> m <f3>	(pel-customize-library &optional OTHER-WINDOW)	Customize Emacs support for cursor and multiple-cursors.
Temporary change the cursor's color	<f11> C-c	(pel-set-cursor-color COLORNAME)	Set cursor to specified COLORNAME string. Prompts for the color name, support color name completion with tab.  Only available in graphics mode.
		<ul style="list-style-type: none">• Ignore the request when color is not a string.• Return the COLOR string on success, nil otherwise.•  When used as an interactive command the new cursor color sticks only until the overwrite-mode is toggled.<ul style="list-style-type: none">•  To make the color change persist, modify the `cursor' or the `pel-cursor-overwrite-mode-color' user options.	
Permanently change the cursor's color See also: 🔗 Customize	<f11> <f2> E C-c	(pel-customize-cursor &optional OTHER-WINDOW)	Quicks access to the customize buffer to set the cursor default color. <ul style="list-style-type: none">• It sets the color permanently if the customization is saved.  Only available in graphics mode.
Multiple Cursors Mode See demo on Emacs-Rocks	With this set of commands you can set multiple cursors in the window to operate on each location simultaneously.  This requires the multiple-cursors external package.  With PEL, set the pel-use-multiple-cursors user-option set to t to install and activate it. There's 2 main methods with this package, both start by identifying the locations of the cursors: <ul style="list-style-type: none">• Make a vertical selection of several lines (on any column) and use the mc/edit-lines command (mapped to <f11> m m) to activate one cursor per line.• Highlight some text and then use the other 3 commands to activate a cursor before the marked area and on the next, previous or all instances of the same text in the buffer:<ul style="list-style-type: none">• mc/mark-next-like-this• mc/mark-previous-like-this• mc/mark-all-like-this There are other methods to set the cursors: <ul style="list-style-type: none">• Another one is to use Visual Regexp (see below).• See also 🔗IMJ- Lispy which supports multiple cursor on potentially different text in multiple locations of Lisp source code. <ul style="list-style-type: none">• Exit the multiple-cursors mode with RET or C-g.  While this is fine and very useful for editing commands be away that every command issued from the buffer with multiple cursor actives will also be potentially applied at the location of each cursor. Therefore if you issue another type of command, like execution via M-x or help request with C-h or <f1> , Emacs will prompt asking whether that command applies to all cursors.		
	Alternatives to multiple-cursor technique: <ul style="list-style-type: none">• The iedit-mode, describe later in this page, can be used to replace all or some instances of selected text like variables, function names, etc...• See Xah Lee comment on this promoting keyboard macros, and other techniques.		
Activate multiple cursors on marked lines	<f11> m m	(mc/edit-lines &optional ARG)	Add one cursor to each line of the active region. Starts from mark and moves in straight down or up towards the line point is on. <ul style="list-style-type: none">• What is done with lines which are not long enough is governed by 'mc/edit-lines-empty-lines'. The prefix argument ARG can be used to override this.<ul style="list-style-type: none">• If ARG negative, short lines will be ignored.• Any other non-nil value will cause short lines to be padded.
Set cursor on next instance of marked text	<f11> m n	(mc/mark-next-like-this ARG)	Find and mark the next part of the buffer matching the currently active region <ul style="list-style-type: none">• If no region is active add a cursor on the next line.• With negative ARG, delete the last one instead.• With zero ARG, skip the last one and mark next.
Set cursor on previous instance of marked text	<f11> m p	(mc/mark-previous-like-this ARG)	Find and mark the previous part of the buffer matching the currently active region <ul style="list-style-type: none">• If no region is active add a cursor on the previous line• With negative ARG, delete the last one instead.• With zero ARG, skip the last one and mark next.
Set cursor on all instances of marked text	<f11> m a	(mc/mark-all-like-this)	Find and mark all the parts of the buffer matching the currently active region.
Visual Regexp to multiple cursors See also: 🔗 Search/Replace	Another way to create multiple cursor is to use the following commands that perform a regular expression search to identify the location of each cursor.  Both require the multiple-cursors external package.  With PEL, set the pel-use-multiple-cursors user-option set to t to install and activate it.  vr/mc-mark requires the visual-regexp external package.  With PEL, set the pel-use-visual-regexp user-option set to t to install and activate it.  vr/select-mc-mark requires the visual-regexp external package.  With PEL, set the pel-use-visual-regexp-steroids user-option set to t to install and activate it.		
Visual Regexp Search to multiple-cursors	<ul style="list-style-type: none">• <f11> s x M• C-c m	(vr/mc-mark REGEXP START END)	Convert regexp selection to multiple cursors. <ul style="list-style-type: none">• First performs a Visual regexp search. When the result of the search is accepted (by hitting RET) all matches are converted to multiple cursors, which allows performing the same operations on all matches until the user quits the multiple cursor operation with C-g.  PEL only activates the C-c m binding if the pel-bind-keys-for-regexp user option is set to t.
Visual Regexp Search to multiple-cursors with engine selection	<f11> s x M-m	(vr/select-mc-mark)	
Highlight Current Line	Highlighting the current line may help to find the cursor when editing with big windows. These commands control line highlighting.		
Toggle line highlight mode See also: 🔗 Highlight	<ul style="list-style-type: none">• <f11> h -• <f11> 0	(hl-line-mode &optional ARG)	Toggle highlighting of the current line (HI-Line mode) in the current buffer. <ul style="list-style-type: none">• With a prefix argument ARG, enable HI-Line mode if ARG is positive, and disable it otherwise.• When same buffer is shown in several windows, the highlighting might show in each of them. Change that with pel-toggle-hl-line-sticky with: <f11> h s  A quick way to find where your cursor is located is to hit <f11> 0 quickly to toggle line highlighting on and off (that key binding is easier to type than the alternative, which exists for consistency, remember that you can use <f1> k to get help for a specific key binding and see all the key bindings for a command, allowing you to discover its other bindings.)
Change color of highlight line for session See also: 🔗 Highlight	<f11> h h	(pel-set-highlight-color COLORNAME)	Set the colour of the highlight line used in the line highlight mode (affects all buffers). <ul style="list-style-type: none">• Prompt for color name, use tab completion to show available colours with their names.• The change does not persist when Emacs is closed. To select a persistent color, then customize the highlight user option (see next row).

Operation	Keystroke	Function	Note
Set highlight color and attributes permanently See also: ⌘ Highlight	<f11> h H	(pel-customize-highlight)	Open the customize buffer to change the highlight user option color and other attributes. <ul style="list-style-type: none"> As with all customizations, you can activate the change for this Emacs session or save it to make it persist across Emacs sessions. With this you can set other attributes such as underlining (which will underline only text present in the buffer, useful to detect end-of-line whitespace), and other attributes.
Toggle line highlighting affecting all windows See also: ⌘ Highlight	<f11> h s	(pel-toggle-hl-line-sticky)	Toggle current line highlight to all windows showing the current buffer or just the current one. <ul style="list-style-type: none"> Toggles the value of 'hl-line-sticky-flag' between t and nil.
Highlight current column	The following command provide a vertical line across the entire window at the cursor location. <ul style="list-style-type: none"> Useful when creating tables or checking indentation manually. vlime also provides the vlime-global-mode to activate the vertical line in all buffers; PEL has no binding for it because it slows Emacs too much. 		
Toggle Vline Mode See also: ⌘ Highlight , ⌘ Hide/Show	<f11> h	(vline-mode &optional ARG)	Toggle the display of a vertical line spanning the entire window at the cursor column.  Requires: vline.el  PEL activates this when pel-use-vline user option is t .
iEdit mode	iEdit Mode - Edit multiple regions in the same way simultaneously  This requires the iedit external package.  PEL downloads, installs it when any of the pel-use-iedit or pel-use-lispy user options is set to t .		
Toggle iedit mode See also: <ul style="list-style-type: none"> ⌘ Search/Replace ⌘ Highlight 	<ul style="list-style-type: none"> C-; <f11> e <f11> h i <f11> m i 	(iedit-mode &optional ARG)	Toggle iEdit mode: edit all symbols in scope or region simultaneously .  Both iEdit and Flyspell use the C-; key as their default binding. <ul style="list-style-type: none"> PEL detects and reports that situation: modify the binding of one of them if you see it.  See ⌘ Search/Replace where all the iedit-mode commands are described.