

## Cursor / Multiple-Cursors

Operation	Keystroke	Function	Note
<b>Controlling Emacs' Cursor</b>  See also: <a href="#">📖 Customize</a>          Last updated on:	You can control Emacs cursor color and shape <b>when Emacs is running in graphical mode</b> only. 👁 Emacs provide the <b>cursor-type</b> customize option to select the default cursor shape. This is part of the <b>display</b> customization group. 🔧 PEL provides the following user options cursor control for Emacs running in graphics mode: <ul style="list-style-type: none"><li><b>pel-cursor-overwrite-mode-color</b>: Selects cursor color in overwrite-mode. Default is black on white background and white on black background.</li><li><b>pel-cursor-type-when-mark</b> : 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 the region.</li></ul> Also, the <b>cursor-chg.el</b> file also exists but I have found it to slow Emacs. Therefore PEL implements its own control.		
Open this PDF file. See also: <a href="#">📖 Help/Info</a>	<b>&lt;f11&gt; m &lt;f1&gt;</b>	( <b>pel-help-pdf</b> &optional OPEN-WEB-PAGE)	Open the <a href="#">📖 Cursor</a> local PDF. If the prefix argument (like <b>C-u</b> or <b>M--</b> ) is used, then it opens the remote GitHub hosted raw PDF instead. If the <b>pel-flip-help-pdf-arg</b> user-option is set it's the other way around.
Customize PEL Cursor control	<b>&lt;f11&gt; m &lt;f2&gt;</b>	( <b>pel-customize-pel</b> &optional OTHER-WINDOW)	Customize PEL support for cursor and multiple-cursors. <ul style="list-style-type: none"><li>If OTHER-WINDOW is non-nil (use <b>C-u</b>) , display in other window.</li></ul>
Customize Emacs cursor control.	<b>&lt;f11&gt; m &lt;f3&gt;</b>	( <b>pel-customize-library</b> &optional OTHER-WINDOW)	Customize Emacs support for cursor and multiple-cursors: provide access to the following customization groups: <ul style="list-style-type: none"><li><b>cursor</b> : where most cursor settings are, including the PEL user-options.</li><li><b>display</b>: where <b>cursor-type</b> is defined. To change default cursor only.</li><li><b>multiple-cursor</b>: for controlling the multiple cursor settings.</li></ul>
<b>Change Cursor</b>	These work in graphical emacs only		
Temporary change the cursor's color	<b>&lt;f11&gt; C-c</b>	( <b>pel-set-cursor-color</b> COLORNAME)  <ul style="list-style-type: none"><li>Ignore the request when color is not a string. Return the COLOR string on success, nil otherwise.</li><li>⚠ When used as an interactive command the new cursor color sticks only until the overwrite-mode is toggled.<ul style="list-style-type: none"><li>👉 To make the color change persist, modify the <b>`cursor'</b> or the <b>`pel-cursor-overwrite-mode-color'</b> user options.</li></ul></li></ul>	Set cursor to specified COLORNAME string. Prompts for the color name, support color name completion with tab. ⚠ <b>Only available in graphics mode.</b>
Permanently change the cursor's color See also: <a href="#">📖 Customize</a>	<b>&lt;f11&gt; &lt;f2&gt; E C-c</b>	( <b>pel-customize-cursor</b> &optional OTHER-WINDOW)	Quicks access to the customize buffer to set the cursor default color. <ul style="list-style-type: none"><li>It sets the color permanently if the customization is saved.</li></ul> ⚠ <b>Only available in graphics mode.</b>
<b>Multiple Cursors Mode</b>          See <a href="#">demo on Emacs-Rocks</a>          💡 Use it to mark multiple areas and operate on all of them simultaneously.          See also: <a href="#">📖 Marking</a>          See also: <a href="#">📖 Windows</a>          See also: <a href="#">📖 Keyboard Macros</a>	With this set of commands you can set multiple cursors in the window to operate on each location simultaneously. 📦 This requires the <a href="#">multiple-cursors</a> external package. 🔧 With PEL, set the <b>pel-use-multiple-cursors</b> 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"><li>Make a vertical selection of several lines (on any column) and use the <b>mc/edit-lines</b> (mapped to <b>&lt;f11&gt; m m</b>) to activate one cursor per line.</li><li>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: <b>mc/mark-next-like-this</b> , <b>mc/mark-previous-like-this</b> , <b>mc/mark-all-like-this</b></li></ul> There are other methods to set the cursors: <ul style="list-style-type: none"><li>Another one is to use Visual Regexp (see below).</li><li>See also <a href="#">🔗📖 Lisp</a> which supports multiple cursor on potentially different text in multiple locations of Lisp source code.</li></ul> ⚠ While this is fine and very useful for some editing commands be aware <b>that every command issued</b> from the buffer with multiple cursor actives will also be potentially applied at the location of each cursor. Therefore if you issue prompting commands, like execution via <b>M-x</b> or help request with <b>C-h</b> or <b>&lt;f1&gt;</b> , Emacs will prompt asking whether that command applies to all cursors. 👉 <b>To cancel a multi-cursor operation</b> you often have to issue <b>C-g</b> twice: once to cancel the text marking and then again to cancel the multi-cursor mode. 👉 The number of matches is shown on the window mode-line with something like "mc:56" identifying 56 matches. 💡 You can use these commands to quickly get a count of matches in the buffer: look at the count displayed in the mode-line. Just remember to cancel. 💡 To see all cursors in a larger area of the current buffer that your screen height can show, split your window with several side-by side ones (with <b>C-x 3</b> ) and use the follow-mode (with <b>&lt;f11&gt; w f</b> ) to line up the windows. Exit follow-mode and then use the multi-cursor command to perform the change. 👉 Once you have identified some matches, use the <b>C-'</b> or <b>&lt;f11&gt; m /</b> key to hide non-matching lines to see more of those matches then proceed with your editing commands. ⚠ Multi-cursor will operate <b>only</b> on the <b>visible</b> part of a buffer for most commands. See the mc match count on the modelling to confirm.		
Alternatives to multiple-cursor technique: <ul style="list-style-type: none"><li>The <b>iedit-mode</b>, describe later in this page, can be used to replace all or some instances of selected text like variables, function names, etc...</li><li>See <a href="#">Xah Lee comment on this</a> promoting the use of keyboard macros and other techniques instead of using multi-cursors.<ul style="list-style-type: none"><li>Personally I like multi-cursor when modifying text inside a single buffer and use keyboard macros when modifying text in a buffer taking data from another or several other windows. I often use multi-cursor <i>and</i> keyboard macros together for even better leverage.</li></ul></li></ul>			
Toggle multiple cursor mode	<b>&lt;f11&gt; m M</b>	( <b>multiple-cursors-mode</b> &optional ARG)	Toggle the 'Multiple-Cursors mode' minor mode. If the prefix argument is positive, enable the mode, and if it is zero or negative, disable the mode. <ul style="list-style-type: none"><li>This is useful in some conditions when you want to disable the multiple cursor mode.</li></ul>
<b>Hide un-matched</b>	Once you have selected matched patterns with the commands below, you can hide the other, non-matched lines, showing only lines with matched area, with some lines before and after. Possibly several such sections separated by special lines. <ul style="list-style-type: none"><li>Hiding these non matching lines help see the modifications over a large number of separated matching lines.</li></ul>		
Hide/show unmatched lines	<b>C-'</b>  <b>&lt;f11&gt; m M-/</b>	( <b>mc-hide-unmatched-lines-mode</b> ARG)	Hide/show all lines that do not have one of the multiple-cursors. <ul style="list-style-type: none"><li>Show some lines before and after.</li><li>If there are several group of matches in several areas, show separating lines between them.</li><li>Issue the command again to restore a normal, complete view to the buffer.</li></ul>
<b>• Set Multiple</b>	<b>Cursors on all instances of guessed area based on position of point</b>		
Mark all from point, repeat to increase number of selections	<b>&lt;f11&gt; m m</b>	( <b>mc/mark-all-like-this-dwim</b> )	Tries to guess what you want to mark all of. <ul style="list-style-type: none"><li>Can be pressed multiple times to increase selection to a larger area.<ul style="list-style-type: none"><li>You can also use <b>&lt;f5&gt;</b> to 'repeat' instead of retyping <b>&lt;f1&gt; m m</b> .</li></ul></li><li>With prefix, it behaves the same as 'mc/mark-all-like-this'</li></ul>
Mark all from point	<b>&lt;f11&gt; m M-m</b>	( <b>mc/mark-all-dwim</b> )	Tries even harder to guess what you want to mark all of. <ul style="list-style-type: none"><li>If the region is active and spans multiple lines, it will behave as if 'mc/mark-all-in-region'.</li><li>With the prefix ARG, it will call 'mc/edit-lines' instead.</li><li>If the region is inactive or on a single line, it will behave like 'mc/mark-all-like-this-dwim'.</li></ul>
Mark more like this at point using cursor navigation	<b>&lt;f11&gt; m .</b>	( <b>mc/mark-more-like-this-extended</b> )	Like mark-more-like-this, but then lets you adjust with arrows key. <ul style="list-style-type: none"><li>The adjustments work like this:<ul style="list-style-type: none"><li><b>&lt;up&gt;</b> Mark previous like this and set direction to 'up'. While going up:<ul style="list-style-type: none"><li><b>&lt;left&gt;</b> Skip past the cursor furthest up</li><li><b>&lt;right&gt;</b> Remove the cursor furthest up</li></ul></li><li><b>&lt;down&gt;</b> Mark next like this and set direction to 'down'. While going down:<ul style="list-style-type: none"><li><b>&lt;left&gt;</b> Remove the cursor furthest down</li><li><b>&lt;right&gt;</b> Skip past the cursor furthest down</li></ul></li></ul></li></ul>
<b>• Set Multiple</b>	<b>Cursors on the some columns of multiple lines.</b> <span>📌 Note: no need to mark lines first</span>		
Mark multiple lines on a column ★ ★ ★ ★	<b>&lt;f11&gt; m c</b>	( <b>set-rectangular-region-anchor</b> )	Anchors the rectangular region at point. Activates <b>rectangular-region-mode</b> . <ul style="list-style-type: none"><li>Think of this one as 'set-mark' except you're marking a rectangular region.</li><li>It is an exceedingly quick way of adding multiple cursors to multiple lines.<ul style="list-style-type: none"><li>Issue the command then move cursor to identify area.</li></ul></li><li>👉 Unaffected by 'void' space on sorter lines! Making this very useful to:<ul style="list-style-type: none"><li>insert or remove indentation after some leading text (like inside a table).</li><li>delete or fill a rectangle of text with any columns of text.</li></ul></li></ul> ⚠ Remember that multi-cursor only operate on the visible part of the buffer.

Operation	Keystroke	Function	Note
<ul style="list-style-type: none"> <li><b>Set Multiple</b></li> </ul>	<b>Cursors on <i>marked</i> lines ...</b>		👉 Note: the lines must be marked first!
... each line on same column	<f11> m l	(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"> <li>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"> <li>If ARG negative, short lines will be ignored.</li> <li>Any other non-nil value will cause short lines to be padded.</li> </ul> </li> </ul>
... at beginning of line	<f11> m C-a	(mc/edit-beginnings-of-lines)	Add one cursor to the beginning of each line in the active region.
... at end of line	<f11> m C-e	(mc/edit-ends-of-lines)	Add one cursor to the end of each line in the active region.
<ul style="list-style-type: none"> <li><b>Set Multiple</b></li> </ul>	<b>Cursors on marked area like this ...</b>		(on the same side as point on currently marked area)
... in buffer	<f11> m a	(mc/mark-all-like-this)	Find and mark all the parts of the buffer matching the currently active region.
... in defun	<f11> m C-M-a	(mc/mark-all-like-this-in-defun)	Mark all like this in defun. ⚠️ The concept of “defun” depends on the major mode. The area actually selected is controlled by the function <b>narrow-to-defun</b> and its support by the current major mode.
... in region	<f11> m ?	(mc/mark-all-in-region)	Find and mark all the parts in the region matching the given search. <ul style="list-style-type: none"> <li>Prompt for string to search inside the marked region.</li> </ul>
<ul style="list-style-type: none"> <li><b>Add more</b></li> </ul>	<b>cursors like the current one(s)..</b>		
... set cursor to next instance of current match	<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"> <li>If no region is active add a cursor on the next line.</li> <li>With negative ARG, delete the last one instead.</li> <li>With zero ARG, skip the last one and mark next.</li> </ul>
... set cursor to previous instance of current match	<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"> <li>If no region is active add a cursor on the previous line</li> <li>With negative ARG, delete the last one instead.</li> <li>With zero ARG, skip the last one and mark next.</li> </ul>
<ul style="list-style-type: none"> <li><b>Remove</b></li> </ul>	<b>cursors like the current one(s)...</b>		
... from the end of selected ones	<f11> m N	(mc/unmark-next-like-this)	Deselect next part of the buffer matching the currently active region.
... from the beginning of selected ones	<f11> m P	(mc/unmark-previous-like-this)	Deselect previous part of the buffer matching the currently active region.
<ul style="list-style-type: none"> <li><b>Skip to ...</b></li> </ul>			
... the next match	<f11> m M-n	(mc/skip-to-next-like-this)	Skip the current one and select the next part of the buffer matching the currently active region.
... the previous match	<f11> m M-p	(mc/skip-to-previous-like-this)	Skip the current one and select the prev part of the buffer matching the currently active region.
<ul style="list-style-type: none"> <li><b>By Word:</b></li> </ul>	<b>Set cursor(s) to the...</b>		
<ul style="list-style-type: none"> <li>Match next instance currently highlighted word, or</li> <li>cursor on next line</li> </ul>	<f11> m w	(mc/mark-next-word-like-this ARG)	Find and mark the next word of the buffer matching the currently active region. <ul style="list-style-type: none"> <li>The matching region must be a whole word to be a match.</li> <li>If no region is active add a cursor on the next line.</li> <li>With negative ARG, delete the last one instead.</li> <li>With zero ARG, skip the last one and mark next.</li> </ul>
<ul style="list-style-type: none"> <li>Match next instance of marked region, or</li> <li>Match current word its next instance</li> </ul>	<f11> m M-w	(mc/mark-next-like-this-word ARG)	Find and mark the next part of the buffer matching the currently active region. <ul style="list-style-type: none"> <li>If no region is active, mark the word at the point and find the next match.</li> <li>With negative ARG, delete the last one instead.</li> <li>With zero ARG, skip the last one and mark next.</li> </ul>
<ul style="list-style-type: none"> <li>Match previous instance currently highlighted word, or</li> <li>cursor on previous line</li> </ul>	<f11> m W	(mc/mark-previous-word-like-this ARG)	Find and mark the previous part of the buffer matching the currently active region. <ul style="list-style-type: none"> <li>The matching region must be a whole word to be a match.</li> <li>If no region is active add a cursor on the previous line.</li> <li>With negative ARG, delete the last one instead.</li> <li>With zero ARG, skip the last one and mark next.</li> </ul>
<ul style="list-style-type: none"> <li>Match previous instance of marked region, or</li> <li>Match current word its previous instance</li> </ul>	<f11> m M-W	(mc/mark-previous-like-this-word ARG)	Find and mark the previous part of the buffer matching the currently active region. <ul style="list-style-type: none"> <li>If no region is active, mark the word at the point and find the previous match.</li> <li>With negative ARG, delete the last one instead.</li> <li>With zero ARG, skip the last one and mark previous.</li> </ul>
Match all words matching: <ul style="list-style-type: none"> <li>word at point, or</li> <li>currently marked area</li> </ul>	<f11> m C-w	(mc/mark-all-words-like-this)	Find and mark all words in the buffer matching the word at point or currently marked.
Match all words like current word, in the current defun	<f11> m C-M-w	(mc/mark-all-words-like-this-in-defun)	Mark all words like this in defun.
<ul style="list-style-type: none"> <li><b>By Symbol:</b></li> </ul>	<b>Set cursor(s) to the...</b>		
<ul style="list-style-type: none"> <li>Match next instance currently highlighted symbol, or</li> <li>cursor on next line</li> </ul>	<f11> m s	(mc/mark-next-symbol-like-this ARG)	Find and mark the next symbol of the buffer matching the currently active region. <ul style="list-style-type: none"> <li>The <b>matching region must</b> be a <b>whole symbol</b> to be a match.</li> <li>If no region is active add a cursor on the next line.</li> <li>With negative ARG, delete the last one instead. With zero ARG, skip the last one and mark next.</li> </ul>
<ul style="list-style-type: none"> <li>Match next instance of marked region, or</li> <li>Match current symbol its next instance</li> </ul>	<f11> m M-s	(mc/mark-next-like-this-symbol ARG)	Find and mark the next part of the buffer matching the currently active region. <ul style="list-style-type: none"> <li>If no region is active, mark the symbol at the point and find the next match.</li> <li>With negative ARG, delete the last one instead. With zero ARG, skip the last one and mark next.</li> </ul>
<ul style="list-style-type: none"> <li>Match previous instance currently highlighted symbol, or</li> <li>cursor on previous line</li> </ul>	<f11> m S	(mc/mark-previous-symbol-like-this ARG)	Find and mark the previous part of the buffer matching the currently active region. <ul style="list-style-type: none"> <li>The matching region must be a whole symbol to be a match.</li> <li>If no region is active add a cursor on the previous line.</li> <li>With negative ARG, delete the last one instead. With zero ARG, skip the last one and mark next.</li> </ul>
<ul style="list-style-type: none"> <li>Match previous instance of marked region, or</li> <li>Match current symbol its previous instance</li> </ul>	<f11> m M-S	(mc/mark-previous-like-this-symbol ARG)	Find and mark the previous part of the buffer matching the currently active region. <ul style="list-style-type: none"> <li>If no region is active, mark the symbol at the point and find the previous match.</li> <li>With negative ARG, delete the last one instead.</li> <li>With zero ARG, skip the last one and mark previous.</li> </ul>
Match all symbols matching: <ul style="list-style-type: none"> <li>symbol at point, or</li> <li>currently marked area</li> </ul>	<f11> m C-s	(mc/mark-all-symbols-like-this)	Find and mark all symbols in the buffer matching the symbol at point or currently marked.
Match all symbols like current symbol, in the current defun	<f11> m C-M-s	(mc/mark-all-symbols-like-this-in-defun)	Mark all symbols like this in defun.
<ul style="list-style-type: none"> <li><b>Insert at ... all cursors</b></li> </ul>			
Insert increasing numbers at each cursor	<f11> m i n	(mc/insert-numbers ARG)	Insert increasing numbers for each cursor, starting at `mc/insert-numbers-default' or ARG. <ul style="list-style-type: none"> <li><b>mc/insert-numbers-default</b> is u user-option in the multiple-cursors customization group that defaults to 0.</li> </ul>
Insert increasing letter at each cursor	<f11> m i l	(mc/insert-letters ARG)	Insert increasing letters for each cursor, starting at 0 or ARG. <ul style="list-style-type: none"> <li>Where letter[0]=a letter[2]=c letter[26]=aa</li> </ul>

Operation	Keystroke	Function	Note
• <b>Sort</b> ...			
Sort marked ares of lines	<f11> m o	(mc/sort-regions)	Sort the marked portion of the lines that have one of the cursors. ⚠ The non-marked areas of the lines affected are <b>NOT</b> moved.
Reverse order of marked area	<f11> m O	(mc/reverse-regions)	Reverse the order of the marked regions. ⚠ The non-marked areas of the lines affected are <b>NOT</b> moved.
• <b>Special</b> ...			
Mark current SGML tag and its pair	<f11> m t	(mc/mark-sgml-tag-pair)	Mark the SGLM tag we're in and its pair for renaming. 👉 Useful for editing SGML, HTML, etc...
Vertical align with spaces See also: <a href="#">↗ Align</a>	<f11> m	(mc/vertical-align-with-space)	Aligns all cursors with whitespace to the one with the highest column number (the rightest). <ul style="list-style-type: none"><li>Keep multiple cursors active, use this to align text while working with the multiple cursors.</li><li>Might not behave as intended if more than one cursors are on the same line.</li></ul>
<b>Visual Regexp to multiple cursors</b>  See also: <a href="#">↗ Search/Replace</a>	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. <ul style="list-style-type: none"><li>Both require the <a href="#">multiple-cursors</a> external package. 📦 With PEL, set the <b>pel-use-multiple-cursors</b> user-option set to t to install and activate it.</li><li>📦 <b>vr/mc-mark</b> requires the <a href="#">visual-regexp</a> external package. 📦 With PEL, set the <b>pel-use-visual-regexp</b> user-option set to t to install and activate it.</li><li>📦 <b>vr/select-mc-mark</b> requires the <a href="#">visual-regexp</a> external package. 📦 With PEL, set the <b>pel-use-visual-regexp-steroids</b> user-option set to t to install and activate it.</li></ul>		
Visual Regexp Search to multiple-cursors	<ul style="list-style-type: none"><li>• &lt;f11&gt; s x M</li><li>• C-c m</li></ul>	(vr/mc-mark REGEXP START END)	Convert regexp selection to multiple cursors. <ul style="list-style-type: none"><li>First performs a Visual regexp search. When the result of the search is accepted (by hitting <b>RET</b>) 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 <b>C-g</b>.</li></ul>
Visual Regexp Search to multiple-cursors with engine selection	<f11> s x M-m	(vr/select-mc-mark)	📦 PEL only activates the <b>C-c m</b> binding if the <b>pel-bind-keys-for-regexp</b> user option is set to t.
<b>Highlight Current Line</b>	Highlighting the current line may help to find the cursor when editing with big windows. These commands control line highlighting.		
<b>Toggle line highlight mode</b>  See also: <a href="#">↗ Highlight</a>	<ul style="list-style-type: none"><li>• &lt;f11&gt; h -</li><li>• &lt;f11&gt; 0</li></ul>	(hl-line-mode &optional ARG)	Toggle highlighting of the current line (HI-Line mode) in the current buffer. <ul style="list-style-type: none"><li>With a prefix argument ARG, enable HI-Line mode if ARG is positive, and disable it otherwise.</li><li>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: &lt;f11&gt; h s</li></ul> 👉 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.)
<b>Change color of highlight line for session</b>  See also: <a href="#">↗ Highlight</a>	<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"><li>Prompt for color name, use tab completion to show available colours with their names.</li><li>The change does not persist when Emacs is closed. To select a persistent color, then customize the <b>highlight</b> user option (see next row).</li></ul>
<b>Set highlight color and attributes permanently</b>  See also: <a href="#">↗ Highlight</a>	<f11> h H	(pel-customize-highlight)	Open the customize buffer to change the <b>highlight</b> user option color and other attributes. <ul style="list-style-type: none"><li>As with all customizations, you can activate the change for this Emacs session or save it to make it persist across Emacs sessions.</li><li>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.</li></ul>
<b>Toggle line highlighting affecting all windows</b> See also: <a href="#">↗ Highlight</a>	<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"><li>Toggles the value of ‘hl-line-sticky-flag’ between t and nil.</li></ul>
<b>Highlight current column</b>	The following command provide a vertical line across the entire window at the cursor location. <ul style="list-style-type: none"><li>Useful when creating tables or checking indentation manually.</li><li>vline also provides the vline-global-mode to activate the vertical line in all buffers; PEL has no binding for it because it slows Emacs too much.</li></ul>		
<b>Toggle Vline Mode</b> See also: <a href="#">↗ Highlight</a> , <a href="#">↗ Hide/Show</a>	<f11> h	(vline-mode &optional ARG)	Toggle the display of a vertical line spanning the entire window at the cursor column. 📦 Requires: <a href="#">vline.el</a> 📦 PEL activates this when <b>pel-use-vline</b> user option is t.
<b>iEdit mode</b> ★ ★	iEdit Mode - Edit multiple regions in the same way simultaneously. 👍 Extremely useful!! 📦 This requires the <a href="#">iedit</a> external package. 📦 PEL downloads, installs it when any of the <b>pel-use-iedit</b> or <b>pel-use-lispy</b> user options is set to t.		
<b>Toggle iedit mode</b>  See also: <ul style="list-style-type: none"><li>• <a href="#">↗ Search/Replace</a></li><li>• <a href="#">↗ Highlight</a></li></ul>	<ul style="list-style-type: none"><li>• C-;</li><li>• &lt;f11&gt; e</li><li>• &lt;f11&gt; h e</li><li>• &lt;f11&gt; m e</li></ul>	(iedit-mode &optional ARG)	Toggle iEdit mode: <b>edit all symbols in scope or region simultaneously</b> . ⚠ Both iEdit and Flyspell use the C-; key as their default binding. <ul style="list-style-type: none"><li>PEL detects and reports that situation: modify the binding of one of them if you see it.</li></ul> ➤ See <a href="#">↗ Search/Replace</a> where all the iedit-mode commands are described.