







Highlighting

Operation	Keystroke	Function	Note
<div>Highlighting</div> <div>See also: 🔗 Customize</div>	<div>Emacs provides different ways of highlighting text as well as lines and columns. It performs highlighting on search operations but can also be instructed to highlight regular expressions, matched delimiters, the current line, the current column, the fill column where automatic text wrapping occurs.</div> <div>🔧 PEL provides the pel-pkg-for-highlight customization group to control some aspect of highlighting. Th user options are:</div> <ul style="list-style-type: none">📦 pel-use-fill-column-indicator activates the fill-column-indicator external package for Emacs versions earlier than 27.1. This highlights a vertical line on the fill-column, the column where automatic line wrapping occurs (when the mode is active).📦 pel-use-vline activates vline.el to create a vertical bar across the entire buffer that moves point (the cursor).📦 pel-use-rainbow-delimiters activates rainbow-delimiters.el that highlights matching parens delimiters.📦 pel-use-iedit activates iedit external package which allows you to interactively edit all symbols inside current scope or marked region. <div>PEL provides a set of key bindings to various highlight control commands that are regrouped under the pel:buffer prefix: <f11> b. They are listed in the table below along the native Emacs key bindings for highlight control.</div>		
<div>Open this PDF file.</div> <div>See also: 🔗 Help/Info</div>	<f11> b h <f1>	(pel-help-pdf &optional OPEN-WEB-PAGE)	Open the local copy of the 🔗 Highlight PDF file unless a command prefix (like C-u) was used. In that case it opens the Github-hosted file instead.
<div>🔗 Customize PEL highlighting control</div>	<f11> b h <f2>	(pel-customize-pel &optional OTHER-WINDOW)	Customize PEL support for buffer highlight management: fill-column-indicator, vline, parinfer, rainbow-delimiters. <ul style="list-style-type: none">If OTHER-WINDOW is non-nil (use C-u), display in other window.
<div>🔗 Customize Emacs highlighting control</div>	<f11> b h <f3>	(pel-customize-library &optional OTHER-WINDOW)	Customize Emacs support for buffer highlight management: auto-highlight, edit, rainbow-delimited, line, fill-column-indicator (for Emacs version earlier than 27.1)
<div>Toggle syntax highlighting in current buffer</div> <div>See also: 🔗 Faces/Fonts</div>	<f11> b h F	(font-lock-mode &optional ARG)	Toggle syntax highlighting in this buffer (Font Lock mode). <ul style="list-style-type: none">With a prefix argument ARG, enable Font Lock mode if ARG is positive, and disable it otherwise.
<div>Highlight Blocks</div>	<div>These commands are used to highlight matching parentheses, but also braces and square brackets. Can be used in any editing mode, not restricted to Lisp only (but very useful in for editing Lisp code).</div>		
<div>Enable/Disable highlighting matching block ().{}[]</div>	<ul style="list-style-type: none"><f11> b h (<f12> M-9<M-f12> M-9	(show-paren-mode &optional ARG)	Toggle visualization of matching parens (Show Paren mode). <ul style="list-style-type: none">With a C-u prefix argument ARG, enable Show Paren mode if ARG is positive, and disable it otherwise. If called from Lisp, enable the mode if ARG is omitted or nil. ➡ The <f12> M-9 and <M-f12> M-9 bindings are available for several programming modes.
<div>Enable/Disable coloured highlight of nested blocks ().{}[]</div>	<f11> b h R	(rainbow-delimiters-mode &optional ARG)	Highlight nested parentheses, brackets, and braces with different colours according to their depth. <ul style="list-style-type: none">Customize the depth and colours with M-x customize-group rainbow-delimiters 📦 Requires: rainbow-delimiters.el
<div>Highlight Modes</div>	<div>Note: All highlighting in a buffer is non persistent: it is forgotten when buffer is killed.</div>		
<div>Enable/Disable Highlight Changes Mode</div>	<f11> b h C	(highlight-changes-mode &optional ARG)	Toggle highlighting changes in this buffer (Highlight Changes mode). <ul style="list-style-type: none">With a prefix argument ARG, enable Highlight Changes mode if ARG is positive, and disable it otherwise.In this mode the most recent changes in the buffer are highlighted.However, when this is enabled, the syntax highlighting does not work properly for the new text.
<div>Highlight Current Line</div> <div>See also: 🔗 Cursor</div>	<div>Highlighting the current line may help to find the cursor when editing with big windows. These commands control line highlighting.</div>		
<div>Toggle line highlight mode</div>	<ul style="list-style-type: none"><f11> b 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.<ul style="list-style-type: none">Change that with pel-toggle-hl-line-sticky with: <f11> b h s <div>👉 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.)</div>		
<div>Toggle line highlighting affecting all windows</div>	<f11> b 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.
<div>Change color of highlight line for session</div>	<f11> b 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).		
<div>🔗 Customize highlight color and attributes permanently</div>	<f11> b 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.		
<div>Highlight current column</div>	<div>The following command provide a vertical line across the entire window at the cursor location.</div> <ul style="list-style-type: none">Useful when creating tables or checking indentation manually.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.		
<div>Toggle Vline Mode</div> <div>See also: 🔗 Cursor</div> <div>🔗 Hide/Show</div>	<ul style="list-style-type: none"><f11> b h <f11> 9	(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 .
<div>Highlight Fill Column</div>			
<div>Toggle ruler line on fill column</div>	<ul style="list-style-type: none"><f11> b h \<f11> 8	<ul style="list-style-type: none">(fci-mode &optional ARG)(display-fill-column-indicator-mode &optional ARG)	Toggle a vertical ruler line shown at the buffer's fill-column. <ul style="list-style-type: none">For Emacs earlier than 27.1, 📦 this requires the fill-column-indicator external package 🔧 activated by PEL use-fill-column-indicator user option set to t.For Emacs 27.1 or later, the built-in display-fill-column-indicator-mode is used.⚠️ fci-mode interferes with pop-up menu displays in terminal-mode, at least with the one used by flyspell-correct-word-before-point: the menu lines become all jagged, they do not line up vertically. The problem does not affect Emacs running in graphics mode.
<div>See also:</div> <ul style="list-style-type: none">🔗 Filling/Justification🔗 Spell Checking			

Operation	Keystroke	Function	Note
Show the ruler See also: » Filling/Justification	<f11> b -	(ruler-mode &optional ARG)	Toggle display of ruler in header line (Ruler mode) in window(s) of current buffer. • With a prefix argument ARG, enable Ruler mode if ARG is positive, and disable it otherwise.
	<ul style="list-style-type: none"> The ruler markers are: <ul style="list-style-type: none"> Current cursor position ! Marks every 5 positions. # Current <i>comment-column</i> value, set by C-x ; It controls where the cursor goes when creating a new comment on a line. ¶ Current <i>fill-column</i> value, set by C-x f, controls where the cursor wraps. G Current <i>goal-column</i> value, controlled by C-x C-n. The horizontal position when going up and down lines via C-p and C-n. If nil, its G symbol will not show on the ruler and it will not have any impact on the navigation commands. See the information about the <code>set-goal-column</code> command in the navigation table. 		
Interactive Highlighting	Highlight Specified Text or Text Matching Regexp (Hi Lock Mode)		
Enable/disable highlight lock mode	<f11> b h L	(hi-lock-mode &optional ARG)	Toggle selective highlighting of patterns (Hi Lock mode). • With a prefix argument ARG, enable Hi Lock mode if ARG is positive, and disable it otherwise. • When text is already highlighted toggling the mode off will remove all highlighting; turning it back on will not restore the highlighting.
Enable highlight lock mode to all buffers	<f11> b h G	(global-hi-lock-mode &optional ARG)	Toggle Hi-Lock mode in all buffers. • With prefix ARG, enable Global Hi-Lock mode if ARG is positive; otherwise, disable it.
Highlight text that match regexp	<ul style="list-style-type: none"> M-s h r <f11> b h r 	(highlight-regexp REGEXP &optional FACE)	Set face of each match of REGEXP to FACE. • Interactively, prompt for REGEXP using ‘read-regexp’, then FACE. Use the global history list for FACE.
Highlight entire line that contain text that matches regexp	<ul style="list-style-type: none"> M-s h l <f11> b h l 	(highlight-lines-matching-regexp REGEXP &optional FACE)	Set face of all lines containing a match of REGEXP to FACE. • Interactively, prompt for REGEXP using ‘read-regexp’, then FACE. Use the global history list for FACE.
Highlight matches of phrase	<ul style="list-style-type: none"> M-s h p <f11> b h p 	(highlight-phrase REGEXP &optional FACE)	Set face of each match of phrase REGEXP to FACE. • Interactively, prompt for REGEXP using ‘read-regexp’, then FACE. Use the global history list for FACE. • When called interactively, replace whitespace in user-provided regexp with arbitrary whitespace, and make initial lower-case letters case-insensitive, before highlighting with ‘hi-lock-set-pattern’.
Highlight symbol found near point	<ul style="list-style-type: none"> M-s h . <f11> b h . 	(highlight-symbol-at-point)	Highlight each instance of the symbol at point. • Uses the next face from ‘hi-lock-face-defaults’ without prompting, unless you use a prefix argument.
Un-highlight	<ul style="list-style-type: none"> M-s h u <f11> b h u 	(unhighlight-regexp REGEXP)	Remove highlighting of each match to REGEXP set by hi-lock. • Interactively, prompt for REGEXP, accepting only regexps previously inserted by hi-lock interactive functions. If REGEXP is t (or if C-u was specified interactively), then remove all hi-lock highlighting.
List Highlighted Text	Write a list of currently highlighted text at point with the first command below, then edit if needed and then use the second command to highlight more of the text. These commands can be used to store the highlighting info in comments for persistency, and later re-highlight them using these comments.		
Insert current highlighting regexp/face pairs in buffer at point in comments.	<ul style="list-style-type: none"> M-s h w <f11> b h w 	(hi-lock-write-interactive-patterns)	Write interactively added patterns, if any, into buffer at point. • Interactively added patterns are those normally specified using ‘highlight-regexp’ and ‘highlight-lines-matching-regexp’; they can be found in variable ‘hi-lock-interactive-patterns’.
Extract regexp/face pairs from comments in current buffer and highlight them.	<ul style="list-style-type: none"> M-s h f <f11> b h f 	(pel-hi-lock-find-patterns)	Add patterns from the current buffer to the list of hi-lock patterns. • Does nothing if the current major mode's symbol is a member of the list hi-lock-exclude-modes.  pel-hi-lock-find-patterns is a thin wrapper over hi-lock-find-patterns that displays a warning if executed when the buffer is not already in hi-lock-mode.
Variables			
hi-lock-file-patterns-policy	Controls whether Hi Lock mode should automatically extract and highlight patterns found in a file when it is visited. Its value can be nil (never highlight), ask (query the user), or a function. If it is a function, hi-lock-find-patterns calls it with the patterns as argument; if the function returns non-nil, the patterns are used. The default is ask. Note that patterns are always highlighted if you call hi-lock-find-patterns directly, regardless of the value of this variable.		
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: • » Cursor • » Search/Replace	<ul style="list-style-type: none"> C-; <f11> e <f11> m i 	(iedit-mode &optional ARG)	Toggle iEdit mode. In iEdit mode you can 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. If you see this warning modify the binding of one of the two user options.
	<ul style="list-style-type: none"> This command behaves differently, depending on the mark, point, prefix argument and variable ‘iedit-transient-mark-sensitive’. If iEdit mode is off, turn iEdit mode on. When iEdit mode is turned on, all the occurrences of the current region in the buffer (possibly narrowed) or a region are highlighted. If one occurrence is modified, the change are propagated to all other occurrences simultaneously. If region is not active, ‘iedit-default-occurrence’ is called to get an occurrence candidate, according to the thing at point. It might be url, email address, markup tag or current symbol(or word). In the above two situations, with digit prefix argument 0, only occurrences in current function are matched. This is good for renaming refactoring in programming. You can also switch to iEdit mode from isearch mode directly. The current search string is used as occurrence. All occurrences of the current search string are highlighted. With a universal prefix argument, the occurrence when iEdit mode is turned off last time in current buffer is used as occurrence. This is intended to recover last iEdit mode which is turned off. If region active, iEdit mode is limited within the current region. With repeated universal prefix argument, the occurrence when iEdit mode is turned off last time (might be in other buffer) is used as occurrence. If region active, iEdit mode is limited within the current region. With digital prefix argument 1, iEdit mode is limited on the current symbol or the active region, which means just one instance is highlighted. This behavior serves as a start point of incremental selection work flow. If iEdit mode is on and region is active, iEdit mode is restricted in the region, e.g. the occurrences outside of the region is excluded. If iEdit mode is on and region is active, with a universal prefix argument, iEdit mode is restricted outside of the region, e.g. the occurrences in the region is excluded. Turn off iEdit mode in other situations.  iedit does not properly disable iedit-mode restoration when desktop is restoring previous session. See  See iedit bug #115 , and my fix for this . <ul style="list-style-type: none"> Until this bug-fix is incorporated PEL implements a fix to ensure that iedit-mode restoration is not done during a desktop restoration. 		

Highlight — References

Topic & Link	Description & Notes
Gnu Emacs - 14.13 - Interactive Highlighting	Describes interactive highlighting, a topic for most of the commands listed above.
The Definitive Guide To Syntax Highlighting	A blog, written in 2104, that provides an overview of the highlighting that can be done with Emacs.