File Management

<u>Operation</u>	<u>Keystroke</u>	Function	<u>Note</u>	
File Handling	Emacs provides a large		E (Emacs documentation uses the term "finding" files for that), saving files searching for files or file	
See also: • ∑ Dired • ∑ Customize	content, displaying directory content, etc These are listed in this table. • The directory editing (dired) commands are mainly listed in the <u>Dired</u> table. • There are also several Emacs internal and external packages that provide useful commands. PEL supports several of them, listed below. • Use Emacs customize system to modify their values to activate, deactivate and modify the behaviour of these packages. • PEL <f11> f key prefix followed by either <f2> to access PEL activation group and <f3> to access the external package customization groups. • Once you have modified the relevant user-option values, apply or save them and then either execute M-x pel-init or restart Emacs. PEL provides integration with the following Emacs built-in libraries or functionalities: • Library ffap • Library recent activated by pel-use-recent to list files recently opened.</f3></f2></f11>			
• <u>∑ Key-Chords</u>	 The <u>fzf.el</u> external pa Automatic file time st Automatic update of It also provides integ key-chord rfc-mode 	The <u>fzf.el</u> external package activated by <u>pel-use-fzf</u> to provide fast fuzzy finder using fzf. Automatic file time stamp update on file save activated by <u>pel-update-time-stamp</u> . Automatic update of copyright notice year on file save activated by <u>pel-update-copyright</u> . It also provides integration with the following external packages when the corresponding PEL user-option is activated: **\frac{\text{key-chord}}{\text{key-chord}} activated by <u>pel-use-key-chord</u> , provides convenient key-chords for some commands. **\frac{\text{rfc-mode}}{\text{ivy/counsel}} activated by <u>pel-use-rfc-mode</u> , provides ability to download and browse <u>lETF RFC</u> documents easily (see <u>RFC editor</u>). **\frac{\text{ivy/counsel}}{\text{ivated}} activated by <u>pel-use-counsel</u> provides completion for some file commands. PEL supports more. See **\frac{\text{Completion/Input}}{\text{ivated}} by <u>pel-use-neotree</u> provides an alternative to **\frac{\text{Dired}}{\text{Dired}} to navigate a file directory. **\frac{\text{ivated by pel-use-treemacs}}{\text{ivated by pel-use-treemacs}} provides project-oriented file directory navigation. See **\frac{\text{X Treemacs}}{\text{Treemacs}}		
Open this PDF file. See also: <u>▼ Help/Info</u>	<f11> f <f1> 1</f1></f11>	(pel-help-pdf &optional OPEN-WEB-PAGE)	Open the <u>Notice File-mngt</u> local PDF. If the prefix argument (like C-u or M) is used, then open remote GitHub hosted raw PDF instead. If pel-flip-help-pdf-arg user-option is set it's the other way around.	
<u>Solution Control of Customize PEL File/</u> Directory Management	<f11> f <f2> 1</f2></f11>	(pel-customize-pel &optional OTHER-WINDOW)	Customize PEL support for file management. • If OTHER-WINDOW is non-nil (use C-u), display in other window.	
<u>S</u> Customize Emacs file management support	<f11> f <f3></f3></f11>	(pel-customize-library &optional OTHER-WINDOW)	Customize Emacs support for file management. Includes the following: files, fzf, recentf, popup-switcher.	
Customize Emacs support for file revert	<f11> f r <f3></f3></f11>	(pel-customize-library &optional OTHER-WINDOW)	Customize Emacs support for file automatic revert management.	
Customize ffap (find file at point)	<f11> f a <f3></f3></f11>	(pel-customize-library &optional OTHER-WINDOW)	Customize Emacs support for management of ffap (find file at point).	
Show file mngt status	<f11> f ?</f11>	(pel-show-filemng-status)	Display status of various file management controls: encoding, resolving relative path method, etc	
Open File in OS application	The following command See: <u>Noticed</u>	d opens file(s) outside Emacs, t	using OS applications registered with the file type.	
Open currently file visited in current buffer with the default OS application.	<f11> f C-o</f11>	(pel-open-buffer-file-in-os- app &optional <u>FNAME</u>)	Open the file in the current buffer with the OS-registered application. If the current buffer holds a HTML file, that's a quick way to open the file in your browser. If the buffer is modified, prompt to save buffer first. In dired-mode buffers, open each marked files in its S-registered applications. Inside a dired-mode buffer you can also type z to open the current file or all selected files.	
Opening file	The following commands are available to open/visit files in Emacs buffers. • For some of them the corresponding ido mode function is also shown. • The command used to 'visit' a file, find-file is Emacs default. It supports Emacs' basic tab completion. Packages that support other completion mechanisms can be installed and activated and then the command uses a different completion mechanism. • Sepection Policy Petron Petron Policy Petron Policy Petron			
See Completion/ Input.	change the completion. • See the Completion	on mechanism dynamically. You on/Input and Customize ta	u can change it without restarting Emacs or event re-executing pel-init.	
Open file-open dialog			steal the lock (with 's'), 2) proceed ('p') to edit the file anyway or 3) quit ('q'). On macOS in graphics mode only: open a file, select the file name via an OS File dialog.	
Open (visit) a file/ directory See also:	#-o • <f11> f f • <m-f11> M-f M-f • C-x C-f</m-f11></f11>	(ns-open-file-using-panel) (find-file FILENAME &optional WILDCARDS)	Prompt for the file or directory name to open. Open the selected file/directory in a buffer with the appropriate mode. For directory, the buffer opens in Dired-mode. • With PEL, the <f11> f f and <m-f11> M-f M-f key bindings are always available, regardless of what completion mechanism is in use. It can be used as a fallback when testing various completion packages. I have seen some of them fail and break Ido. • Note that <m-f12> M-f M-f is also available in some major modes to open files in a way that</m-f12></m-f11></f11>	
• <u></u> Customize		(ido-find-file)	takes the major mode into account, like providing a list of files in the project. See major mode pages. Same as above with Ido completion. See <u>Sompletion/Input</u> for available completion modes. The ido-use-filename-at-point user-options control whether ido-find-file uses the file name at	
 Prevent Ido expansion with C-j Open in read-only Change input completion method ► 	point as the basis for selecting the file name to open. Use <f11> f M to dynamically change it • find-file is the original command and uses Emacs default completion. When Ido is used, the ido-find-file command is used instead. • When ido mode is used, you can also: • Type C-f or C-x f to change to original find-file mode and prevent Ido completion from trying to provide the name of an existing file when you want to specify the name of a file that does not exists yet. • Type C-j to accept the file/directory name verbatim without replacement or suggestion. Also useful to open a directory in dired mode. • To open a file in read-only mode you can: Use one of the commands below (C-x C-r, etc) Use C-x C-f then type C-x C-q to change the mode of the buffer to read-only mode. Use <f11> M-c <f4> to select another input completion method. See Sompletion/Input.</f4></f11></f11>			
Open file via popup menu	<f11> f M-f</f11>	(pel-psw-navigate-files)	Open file from a pop-up menu listing files in current directory. Uses (psw-navigate-files "."). Narrow menu list by typing part of the file name. You can also select directory names. Requires popup-switcher PEL activates when pel-use-popup-switcher is t.	
Open another file in buffer	C-x C-v	(find-alternate-file FILENAME &optional WILDCARDS) (ido-find-alternate-file)	Kills buffer and open the newly specified file in a new buffer same window. When ido-mode is used, the ido-find-alternate-file is used instead. Useful when just selected an empty file just selected by mistake.	
Open file in other window	• C-x 4 f • <f11> f o</f11>	(find-file-other-window FILENAME &optional WILDCARDS) (ido-find-file-other-	Edit file FILENAME, in another window. • Like C-x C-f, but creates a new window or reuses an existing one.	
Open file in other frame	C-x 5 f	window) (find-file-other-frame FILENAME &optional WILDCARDS) (ido-find-file-other-frame)	Edit file FILENAME, in another frame. • Like C-x C-f, but creates a new frame or reuses an existing one.	

<u>Operation</u>	<u>Keystroke</u>	Function	<u>Note</u>	
Open same file in other directory	• <f11> f M-d • <m-f11> M-f</m-f11></f11>	(pel-open-file-in-other-dir)	Open file of same name as current one present in another directory. Use it to create clone of current file, or to compare files located inside 2 different VCS dir trees.	
	M-d	Use the prompt to select the	of the directory of currently visited file using the default completion mechanism ('ido' by default). The name of the other directory (which must already exist).	
Set whether ido-find-file uses filename at point	<f11> f M</f11>	(pel-set-ido-use-fname-at- point &optional GLOBALLY)	th without completion. Select dir name, hit <ret> to open the same file in the selected other directory. Set Ido's ability to use the filename at point as a starting point in the current buffer or globally. It can set it to one of: . disabled : don't use filename at point. • guess : try to identify an exiting file name from the name at point. • literal : use name at point in the Ido search for a file name.</ret>	
See also: <u>Completion/</u> Input	 By default this commands sets ido-find-file behaviour for the current buffer only by setting a ido-use-filename-at-point buffer local variable. Use any prefix argument (eg. C-u) to modify the behaviour globally for the current Emacs session, it does not persist across Emacs sessions. For a persistent behaviour change you must customize ido-use-filename-at-point user-option variable. For that, use M-x customize-option. This affects the behaviour of all commands opening file using Ido completion: ido-find-file as the others. 			
Open file at point	The following commands, open files from the file name taken at point (the cursor location). They work regardless of the current input completion method. Note that when using the Ido completion mode, it is possible to instruct Ido to use a file name at point as the basis for the file name to open. This Ido behaviour is controlled by the ido-use-filename-at-point user-option. With PEL you can control it globally or locally with <f11> f M</f11>			
Set base directory for pel-open-at-point	<f11> f ;</f11>	(pel-set-open-at-point-dir)	Set the behaviour of 'pel-open-at-point' in current buffer. Which defaults to value selected by pel- open-file-at-point-dir user-option.	
relative file names		Use visited file parent dieUse buffer's current work	ermine the directory from which a relative file name is built from following methods: rectory (the default).	
Open file or web-page whose name is at point	• <m-f6> • <f11> f .</f11></m-f6>	(pel-open-at-point &optional N)	Open the file, library or the URL, named at point, with potential line & column #s. • If necessary will search source code files in current project as specified by pel-filename-at-point-	
** Command is generic and is also specialized for: • M reStructuredText	• 6y	aoptional Ny	finders user-option. Type <f12> <f4>? to show used file search method in supporting modes. Supports glob characters, partial directory path. When multiple files are found it prompts using the method selected by pel-prompt-read-method user-option. Fig. Rey-chord is available if pel-use-key-chord is non-nil. See Key-Chords.</f4></f12>	
• <u>का - C</u>	S This command work	s generically but is also specia	alized for some major modes, like C, C++, Erlang, reStructuredText, shell scripts.	
• <u>野〔 - C++</u> • <u>野〔 - Erlang</u>	 When executed from the second of the second o	re pages for the major mode sp om with a buffer in sh-mode, th	pecific features. e '=' and ':' characters are used as additional delimiters. Shell variables (such as \$HOME) are	
• 野 - UNIX Shell Delimiting characters ►	In general the command The generic mode ex	expanded. In general the command extracts the file or directory name, and possibly line and column numbers, from text at point and tries to open the file or directory. The generic mode extraction works by identifying the beginning & end of the file/directory/library/URL name string by delimiter characters, one of: tab, newline and: "`' () [] {} <> '''''		
	If embedded space(s) are allowed in the name, point must be located at the first of the 2 delimiter chars. Otherwise point can be anywhere in the name. The name may include glob characters (but not in C/C++ in #include "" or #include <> statements).			
File identification heuristic		The command uses a URL unchanged but uses the following heuristic to identify the exact location of the file/directory:		
<f11> f <f2> w <f11> f ; w</f11></f2></f11>	 In the file/dir name is an absolute path it uses that. Otherwise it builds a absolute path using the extracted relative path name inside the directory identified by the pel-open-file-at-point-dir user-option, which can be 1) use parent directory of currently visited file, or use current working directory, 2) use current working directory, or 3) use user-specified directory. It uses the found file/dir name if it exists. Otherwise it searches for the relative file/dir name in directory tree under the root marker file identified by the pel-project-root-identifiers user-option which is something like .git, .hg, .project, .pel-project (the default). If it can find such a file in the above directories it searches the tree under the found root. If it finds several files it prompts using the current completion mode to allow selection of the appropriate name (see below) and opens the selected one. If it finds only one it opens that file. Otherwise, it prompts showing the name searched and provide the following choices: 1) create the file with specified name, 2) edit the name to search again, 3) use the name found and search for an Emacs library file with that name, or 4) quit. 			
	The command opens the extracted name according to this heuristic: If the string is a properly formatted URL , it opens it using the OS default browser (even if a optional numeric argument specified otherwise), otherwise if the string is a file or directory name it opens it. If the file name is followed by line and column numbers the point is moved to that position in the buffer.			
Select multi-file selection method	When finding several file names, the command lists them and prompts using the method selected by pel-prompt-read-method user-option. The default is a very primitive function implemented by PEL. You can select a more powerful <u>ivy</u> prompting instead. With ivy selected, PEL will automatically set pel-use-ivy to and ivy mode will be installed automatically when you restart Emacs. Note that the command shows all files found by the specified search method, it does not only use the first one found. Use this to detect potential duplication in header file names in large include paths.			
Select target window 🖛	The command opens the file in the window selected by the following logic controlled by presence or absence of typed numerical prefix arguments: Select target window: Without argument: If file or directory is already opened in a window, move point to that window and to the line column coordinates if specified. If no window holds that file, select the target window according to the number of editable windows in frame: if 1, split that window and use the new			
N>20 : open the directory ►	 With <u>prefix numeri</u> N < 0 : create a (abs N) > 20: the 	new window and use that. en open the directory instead	e, use the current window. of the file. Interpret the window position from the N value adjusted: N-20 (or N+20 if N is negative)	
	 N = 0: use the 'other' (the next) window. N = 1, 3, 7or above (excluding 8, 9 and 10): select the target window based on the number of editable windows in frame: if 1 window: split that window and use the new window, if 2 windows: use the other window, if 3 or more windows: use the current window. 			
See function docstring for more info.	 N is: 8: up, 2: down, 4:left, 5:current, 6:right. N is 9: force opening the file in the OS associated application (with N=29 or N=-29, open the file's directory with the OS associated application (eg. macOS Finder, Windows Explorer). If this is a URL, open it in the OS default web browser. Selecting Minibuffer, inexistent or dedicated window is not allowed. 			
Open filename at point in a browser See also: <u>S Key-Chords</u> , <u>S Web</u>	• <f11> f / • <u>6u</u></f11>	(pel-browse-filename-at- point)	Open the file name (or URL) at point inside the system's web browser. • If point is at dir name, open the OS app. browsing dirs (eg. macOS Finder, Windows Explorer). • This is the same as using pel-open-at-point with the argument N set to 9. It is easier to type and PEL assigns its own key-chord for it.	
Open URL at point in a browser See also: • <u>> Key-Chords</u> , • <u>> Web</u>	• <f11> f M-/ • <u>7u</u></f11>	(browse-url-at-point &optional ARG)	Ask a WWW browser to load the URL at or before point. Variable 'browse-url-browser-function' says which browser to use. With prefix argument inverts the value of the option 'browse-url-new-window-flag'. Use <f11> <f2> E u to open the browse-url group that contains relevant user options.</f2></f11>	
Copy URL at point in temporary file and visit the file	<f11> f M-u</f11>	(pel-open-url-at-point)	Copy the URL at point to a local temporary file and visit that file. • A The download copy of the file does not have the same name and may not open with the proper mode because it won't have an extension. The HTML formatted files will be recognized by Emacs but most of the files won't be. • Save the file somewhere else using the C-x C-w key sequence and identify the proper extension to activate the required major mode.	
With goto-address-mode	C-c C-f		This binding is only available when point is over the URL and the goto-address-mode minor mode is active. Use <f11> f u or <f11> f U to activate this mode.</f11></f11>	

<u>Operation</u>	<u>Keystroke</u>	Function	<u>Note</u>	
ffap commands			et. The ffap command is similar to pel-find-file-at-point-in-window but does not support line and	
	however support other	facilities and can be installed to	replace the behaviour of standard file management command bindings such as C-x C-f.	
	key bindings shown bel	ow.	the pel-use-ffap user option is set to either t or to ffap-bindings . In both cases these activate the	
	 When pel-use-ffap is set to ffap-bindings, then PEL also activates the standard ffap bindings which take over the behaviour of the main file finding at dired commands. This means that Ido, Ivy or Helm are no longer available for these commands. If pel-use-ffap is only set to t then the standard ffap bindings is not activated. 			
Find file/URL at point	<f11> f a p</f11>	(ffap &optional FILENAME)	Find FILENAME, guessing a default from text around point.	
			 If 'ffap-url-regexp' is not nil, the FILENAME may also be an URL. Web URL opens in browser. With a prefix, this command behaves exactly like 'ffap-file-finder'. 	
			 If 'ffap-require-prefix' is set, the prefix meaning is reversed. See also the variables 'ffap-dired-wildcards', 'ffap-newfile-prompt', 'ffap-url-unwrap-local', 'ffap- 	
Find file/URL at point -	<f11> f a P</f11>	(ffap-read-only)	url-unwrap-remote', and the functions ffap-file-at-point' and 'ffap-url-at-point'. Like 'ffap', but mark buffer as read-only.	
read only Find another file/URL at	<f11> f a v</f11>	(ffap-alternate-file)	Like 'ffap' and 'find-alternate-file': kills current buffer and open new file in the same window.	
point in window		,		
Find file/URL in other window	<f11> f a w</f11>	(ffap-other-window)	Like 'ffap', but put buffer in another window.	
Find file/URL in other frame	<f11> f a f</f11>	(ffap-other-frame)	Like 'ffap', but put buffer in another frame.	
Find file/URL in other window - read only	<f11> f a W</f11>	(ffap-read-only-other- window)	Like 'ffap', but put buffer in another window and mark as read-only.	
Find file/URL in other frame - read only	<f11> f a F</f11>	(ffap-read-only-other-frame)	Like 'ffap', but put buffer in another frame and mark as read-only.	
Start Dired with file at point	<f11> f a d</f11>	(dired-at-point &optional FILENAME)	Start Dired, defaulting to file at point. See 'ffap'.	
Start Dired with file at point in other window	<f11> f a D</f11>	(ffap-dired-other-window)	Like 'dired-at-point', but put buffer in another window.	
Start Dired with file at point in other frame	<f11> f a M-d</f11>	(ffap-dired-other-frame)	Like 'dired-at-point', but put buffer in another frame.	
List directory of file at	<f11> f a l</f11>	(ffap-list-directory)	Like 'dired-at-point' and 'list-directory'.	
Open a menu of all files,	<f11> f a m</f11>	(ffap-menu &optional	Put up a menu of files and URLs mentioned in this buffer. Set mark, jump to choice, and try to fetch	
URL in current buffer.		RESCAN)	it. The menu is cached in 'ffap-menu-alist', and rebuilt by 'ffap-menu-rescan'.With prefix argument: forces a rebuild. Searches with 'ffap-menu-regexp'.	
Recently opened			EL ensures that Emacs remembers the list of recently opened files and provides: es which function use used to open the recently opened files:	
∑ Completion/Input	ido-recentf-opencounsel-recentf		t or Ido enhanced mechanism. Use <f11> M-c? to list them and see which one is active. Requires counsel external package activated by pel-use-counsel</f11>	
	psw-switch-recent	f: uses a popup menu es a File->Open Recent menu		
Open recently opened files, using active	<f11> f M-r M-r</f11>	(pel-find-recent-file)	Open the recent file prompt using the currently active function.	
method	 The function is selected by pel-initial-recent-f-function. Change with pel-select-recentf-function, bound to <f11> f M-r M-F</f11> When basic Ido is used, type <tab> to get possible expansions listed in a separate buffer.</tab> Ido completion is selectable. Use <f11> M-c ? to list them and see which one is active.</f11> When counsel-recentf is used, you can type C-c C-o to copy the list of files inside a special buffer. 		be <tab> to get possible expansions listed in a separate buffer. ble. Use <f11> M-c ? to list them and see which one is active.</f11></tab>	
Display the name of the function used to prompt for recently opened file	<f11> f M-r M-?</f11>	(pel-show-recentf-function &optional AFTER- SELECTION-P)	Display what function is used to visit recently opened files. • The argument is for internal use, it is not available interactively.	
Select the function used to list/prompt the recently opened files.	<f11> f M-r M-R</f11>	(pel-select-recentf- function & optional RECENTF-FUNCTION SILENT)	Select the function to visit recently opened files. Modifies what is used in the current editing session, not the persistent value selected by the pel-initial-recent-f-function user-option. • The arguments are for internal use, they are not available interactively.	
Edit list of recently opened files	<f11> f M-r M-e</f11>	(recentf-edit-list)	Show a dialog to delete selected files from the recent list. Use this to remove some of the files from the list.	
Open a recently opened	<f11> f M-r M-z</f11>	(fzf-recentf)	Open a recently opened file selected by fzf search.	
file searched by fzf	The following agreement	de open files in road only media	■ Requires the <u>fzf.el</u> external package activated by <u>pel-use-fzf</u> . While in read-only mode, use Use C-x. C-x. to permit editing.	
Open in read-only Open a file in read-only	C-x C-r	(find-file-read-only mode	. While in read-only mode, use Use C-x C-q to permit editing. Edit file FILENAME but don't allow changes.	
mode	U-X U-1	FILENAME & optional WILDCARDS)	Like C-x C-f , but marks buffer as read-only. Use C-x C-q to permit editing.	
Open file in other	• C-x 4 r	(ido-find-file-read-only) (find-file-read-only-	(find-file-read-only-other-window FILENAME &optional WILDCARDS)	
window in read-only mode	• <f11> f 0</f11>	other-window FILENAME &optional WILDCARDS) • (ido-find-file-read-only- other-window)	Edit file FILENAME in another window but don't allow changes. Like C-x C-f , but marks buffer as read-only. Use C-x C-q to permit editing.	
Open as root	On Unix/Linux/macOS s	some files are write protected a	and can only be opened with root privilege with su or sudo. Use the following command for those.	
Open file with root privilege	<f11> f R</f11>	(pel-edit-as-root &optional ARG)	Open a file as root with sudo. Prompt for password if necessary. If already visiting a file and a prefix ARG is specified then edit currently visited file as root.	
Open Literally	Open a file with no encoding conversion: file is opened in the Fundamental mode: the major mode normally associated with the file type is not used. Note that when using Ido completion, it is possible to use a command during completion to force Ido to open the file literally. However, if you are using Emacs default completion, the following command is the only way to open a file literally.			
Visit a file literally: with no encoding support	<f11> f M-1</f11>	(find-file-literally FILENAME)	Visit file FILENAME with no conversion of any kind.	
and conversion	Format conversion and character code conversion are both disabled, and multibyte characters are disabled in the resulting buffer.			
	 The major mode used is Fundamental mode regardless of the file name, and local variable specifications in the file are ignored. Automatic uncompression and adding a newline at the end of the file due to 'require-final-newline' is also disabled. If Emacs already has a buffer which is visiting the file, this command asks you whether to visit it literally instead. 			
Open binary	Open a file in hex binary mode. There are also commands to convert current buffer to hexadecimal editing, like nhexl (described in <u>Natural and the like of the li</u>			
Open a file in hexl-mode	<f11> f M-x</f11>	(hexl-find-file FILENAME)	Edit file FILENAME as a binary file in hex dump format, using the 'hexl-mode'. • Switch to a buffer visiting file FILENAME, creating one if none exists.	
See also: <u>Nauffers</u>			- Switch to a buller visiting life FILENAIVE, creating one if none exists.	

<u>Operation</u>	<u>Keystroke</u>	Function	<u>Note</u>
Fuzzy File Finders	 install and configure the <u>fzf.el</u> ext 	the fzf command line utility, ternal package	inder that can be used within Emacs via the fzf.el emacs front-end. To use it inside Emacs, you must: and use one of the following package to use the corresponding commands: tivated by pel-use-fzf . The fzf commands below are available when this is active. ivated by pel-use-counsel . The counsel commands below are available when this is active.
Open file searched by fzf in current directory	• <f11> M-z M-z • <f11> f z</f11></f11>	(fzf)	Open a file selected by fzf session in the current directory. Type partial file name, use fzf filter expressions. Select one file and hit return to open it inside current window.
Open file searched by fzf in specified directory	• <f11> M-z M-d • <f11> f d</f11></f11>	(fzf-directory)	Prompt for a directory to perform the fzf file search, then open selected file inside current window. Directory prompt uses current completion mode. See Completion/Input .
Open fzf searched file in current or specified directory using ivy I/F	<f11> f c</f11>	(counsel-fzf &optional INITIAL-INPUT INITIAL- DIRECTORY FZF-PROMPT)	Open a file selected by ivy-style prompt using a fzf shell command. • With C-u prefix argument first prompts for the directory to perform the fzf search. • Much slower than (fzf) for large directories because counsel captures fzf output before showing it.
Switch buffer with fzf	<f11> b z</f11>	(fzf-switch-buffer)	Switch buffer in current window by selecting it with fzf. • Uses the fzf command line utility for fast & flexible search. Requires the fzf.el external package activated by pel-use-fzf.
Search/open Git repo member files with fzf	<f11> f g</f11>	(fzf-git-files)	Search Git repository member files in current Git repository with fzf and open user selected file.
Search/open file in Git repo directory tree with fzf	<f11> f G</f11>	(fzf-git)	Search all files in current Git repository with fzf and open user selected file.
Search/open file in Mercurial repo directory tree with fzf	<f11> f H</f11>	(fzf-hg)	Search all files in current Mercurial repository with fzf and open user selected file.
Search/open file in current projectile project with fzf.	<f11> f <f8></f8></f11>	(fzf-projectile)	Search all files in current projectile project with fzf and open selected file. Requires the fzf.el external package activated by pel-use-fzf
See <u>∑</u> Projectile Grep search files with	<f8> M-z</f8>	(fzf-grep)	Requires the <u>projectile</u> external package activated by pel-use-projectile Prompt for string to search and file grep selection expression, show grep results in a fzf session,
fzf for specified regex	_		select appropriate line to open the specific file at appropriate line.
Grep search files with fzf for specified regex in specified directory	<f11> g S</f11>	(fzf-grep-in-dir)	Prompt for directory, then for string to search and file grep selection expression, show grep results in a fzf session, select appropriate line to open the specific file at appropriate line.
Grep search Git repo member files with fzf for specified regex	<f11> g G</f11>	(fzf-git-grep)	Prompt for string to search and file grep selection expression, show grep results over current Git repo searched in a fzf session, select appropriate line to open the specific file at appropriate line. This command does not seem to work properly, it searches but does not always open the file.
Open <u>Dired</u> (Directory Editor) See also: <u>∑ Dired</u>	directory path to C-x (Prompt input comple	Z-f then Dired-mode is used. tion can be changed for these.	uffer in Dired mode, that looks like a ls -I output, which allows several operations. If you specify a You can also use the following commands to open buffer in Dired mode. See <u>Sempletion/Input</u> of file tree browsers, like <u>NeoTree</u> and <u>ztree</u> (see below), or with <u>Sepeedbar</u> .
Open a directory editor	• C-x d • %-D	(dired DIRNAME &optional SWITCHES) (ido-dired)	Opens a Dired-mode buffer on the specified directory. Prompt for the directory name. PEL activates ido when the pel-use-ido-mode user option is set to t.
Run Dired in other window	C-x 4 d	(dired-other-window)	Opens a Dired-mode buffer on the specified directory inside another window. • Prompt for the directory name.
List Directory	C-x C-d	(list-directory DIRNAME &optional VERBOSE)	Display a list of files in or matching DIRNAME, a la 'ls'. • DIRNAME is globbed by the shell if necessary. • Prefix arg (C-u) means supply -l switch to 'ls'.
Jump to file entry in dired buffer ★★ Leaves point on the file jumped to, allowing immediate Dired action, eg.: C-x C-j R renames the file.	C-x C-j	(dired-jump &optional OTHER-WINDOW FILE- NAME)	Jump to Dired buffer corresponding to current buffer. If in a file, Dired the current directory and move to file's line. If in Dired already, pop up a level and goto old directory's line. In case the proper Dired file line cannot be found, refresh the dired buffer and try again. When OTHER-WINDOW is non-nil, jump to Dired buffer in other window. When FILE-NAME is non-nil, jump to its line in Dired. Interactively with prefix argument, read FILE-NAME.
Activating URLs to browse and open files	Emacs provides the goto-url-mode and the goto-url-prog-mode that turn URLs found in the current buffer into clickable buttons. Once the mode is active the following key sequences are available wheel point is over a URL button: C-c RET or the mouse to click on the button. If the URL is an email address a buffer to write an email to that address opens. If the URL is a web or FTP address the system browser is invoked to open the address. C-c C-n: move point to the end of the next URL in the buffer. C-c C-p: move point to to the previous URL in the buffer. C-c C-f: download the file identified by the URL into a local temporary file and visit the file. See (pel-open-url-at-point) above.		
Toggle goto-address- mode	<f11> f u</f11>	(goto-address-mode &optional ARG)	Minor mode to buttonize URLs and e-mail addresses in the current buffer. With a prefix argument ARG, enable the mode if ARG is positive, and disable it otherwise.
Toggle goto-addrress- prog-mode	<f11> f U</f11>	(goto-address-prog-mode &optional ARG)	Like 'goto-address-mode', but only for comments and strings.
Open the URL (email or web page)	C-c RET	(goto-address-at-point &optional EVENT)	Open the URL at point: If URL is a web page: open it in a browser If URL is a mail address: Send mail to address at point: Find e-mail address around or before point. Then search backwards to beginning of line for the start of an e-mail address. If no email address is found there, then load the URL at or before point.
Move to end of next URL in buffer See also: Navigation	C-c C-n <f6> C-n</f6>	(pel-goto-next-url)	Move point forward to the end of the next URL located in the current buffer. • The global <f6> C-n key binding activates the goto-address-mode if it is not already active.</f6>
Move to beginning of previous URL in buffer See also: Navigation	C-c C-p <f11> C-p</f11>	(pel-goto-previous-url)	Move point backward to the beginning of the previous URL located in the current buffer. • The global <f6> C-p key binding activates the goto-address-mode if it is not already active.</f6>
Insert text of another file at point	_	ds can be used to insert text from	om other files at point in the current buffer.
Insert file at point	• C-x i • <f11> f i</f11>	(insert-file FILENAME) (ido-insert-file)	Insert contents of file FILENAME into buffer after point. • Set mark after the inserted text.
Insert file literally at point	<f11> f I</f11>	(insert-file-literally FILENAME)	Insert contents of file FILENAME into buffer after point with no conversion. • Set mark after the inserted text.

<u>Operation</u>	<u>Keystroke</u>	Function	<u>Note</u>
Write text into specified file	The following command	ds can be used to write text se	lected from current buffer into specified file.
Write region text to file	<f11> f w</f11>	(write-region START END	Write current region into specified file.
		FILENAME & optional APPEND VISIT LOCKNAME MUSTBENEW)	Prompts for the specified file.
Append region text to file	<f11> f W</f11>	(append-to-file START END FILENAME)	Append the contents of the region to the end of file FILENAME. • Prompts for the specified file.
Set file mode	<f11> f m</f11>	(set-file-modes FILENAME MODE)	Set mode bits of file named FILENAME to MODE (an integer). Only the 12 low bits of MODE are used. Prompts for file name and then for chmod-like file mode value.
Reverting Files	If you want to use Envariable to a list of regularity	nacs to monitor the content of a gular expressions describing the	to refresh the Emacs buffer visiting that file, you need to "revert" the file. a file that is continuously modified by an external process (like a log file) set the <i>revert-without-query</i> le field it'll apply to. urrent buffer or globally and restart its timer.
Revert a buffer	• <f11> f r f • %-u</f11>	(revert-buffer &optional IGNORE-AUTO	Replace current buffer text with the text of the visited file on disk. • This undoes all changes since the file was visited or saved.
See also: <u>∑ Diff & Merge</u>	50- u	NOCONFIRM PRESERVE- MODES)	 With a prefix argument, offer to revert from latest auto-save file, if that is more recent than the visited file. This is also the command to use to reload a file that was modified on the file system. You can use ediff-current-file to see the difference between the buffer and its disk file. PEL binding for this is <f11> e b f.</f11>
Toggle auto-revert mode	<f11> f r a</f11>	(auto-revert-mode &optional ARG)	Toggle reverting buffer when the file changes (Auto-Revert Mode). With a prefix argument ARG, enable Auto-Revert Mode if ARG is positive, and disable it otherwise. • Auto-Revert Mode is a minor mode that affects only the current buffer. When enabled, it reverts the buffer when the file on disk changes. • When a buffer is reverted, a message is generated. This can be suppressed by setting 'auto-revert-verbose' to nil.
Toggle auto-revert tail mode	• <f11> t • <f11> f r t</f11></f11>	(auto-revert-tail-mode &optional ARG)	Toggle reverting tail of buffer when the file grows. • With a prefix argument ARG, enable Auto-Revert Tail Mode if ARG is positive, and disable it
See also: Scrolling	1222	,	 otherwise. When Auto-Revert Tail Mode is enabled, the tail of the file is constantly followed, as with the shell command 'tail -f'. This means that whenever the file grows on disk (presumably because some background process is appending to it from time to time), this is reflected in the current buffer. You can edit the buffer and turn this mode off and on again as you please. But make sure the background process has stopped writing before you save the file!
Cancel/restart auto- revert timer	<f11> f r SPC</f11>	(pel-auto-revert-set-timer)	Restart or cancel the timer used by Auto-Revert Mode. If such a timer is active, cancel it.
- COUNTY TAINED	 Start a new timer if Global Auto-Revert Mode is active or if Auto-Revert Mode is active in some buffer. Restarting the timer ensures that Auto-Revert Mode will use an up-to-date value of 'auto-revert-interval' (which is normally 5 seconds by default). 		
Saving Files	Use the following comm	nands to save the content of a	buffer to a filesystem file.
	 PEL supports the following controllable actions on file save. Each of these actions are activated via an action-specific PEL user-option, and can temporarily be disabled with a command for the file in the current buffer. The actions and their associated user-option and command are listed here: Action Delete trailing space and lines on save override it for some major modes: Update time stamp on save Update to pel-update-time-stamp Update copyright notice on save Update copyright notice on save Each of these actions are activated via an action-specific PEL user-option, and can their associated user-option and command are listed here: Mey Sequence pel-delete-trailing-whitespace pel-toggle-delete-trailing-space-on-save f11> M-T Update copyright notice on save pel-update-copyright pel-toggle-update-copyright-on-save f11> M-C		
Save file to disk	• C-x C-s • %-s	(save-buffer &optional ARG)	Save current buffer to associated file. By default, it makes the previous version into a backup file if previously requested or if this is the first save. • With C-u: marks this version to become a backup when the next save is done • With C-u C-u: makes the previous version into a backup file • With C-u C-u: marks this version to become a backup when the next save is done, and makes the previous version into a backup file. • With prefix 0: never make the previous version into a backup file. • With prefix 0: never make the previous version into a backup file. • On macOS in graphics mode only: %—s brings a OS file-save dialog. • Save and activated on-file-save actions only occur when the buffer is in "changed" status. Use M—r to flip that status to force an action when it has just been activated.
Save all/some files	C-x s	(save-some-buffers &optional ARG PRED)	Prompt for files that are modified. Options: • y : save • n : don't save • C-r : look at the buffer in question • d : view differences with diff-buffer-with-file
Write buffer to specified file Save As	C-x C-w	(write-file FILENAME &optional CONFIRM) (ido-write-file)	Similar to "Save-As": prompt for the filename. • Can also be yanked in the mini buffer, use M−n to edit it. Use that command to rename the file.
Changed current buffer changed state	M-~	(not-modified &optional ARG)	Mark current buffer as unmodified, not needing to be saved. • With C - u prefix ARG, mark buffer as modified, so C - x C - s will save.
Toggle copyright update on save	<f11> M-@</f11>	(pel-toggle-update- copyright-on-save &optional GLOBALLY)	Toggle copyright update on file save and display current state. • By default change behaviour for local buffer only. • When GLOBALLY argument is non-nil, using any prefix argument, change it for all buffers for the current Emacs editing session (the change does not persist across Emacs sessions). • To modify the global state permanently modify the customized value of the 'pel-update-copyright' user option via the 'pel-pkg-for-filemng' group customize buffer with <f11> f <f2> 1. ☑ This command is only available when the pel-update-copyright is set to t.</f2></f11>
Toggle timestamp update on save	<f11> M-T</f11>	(pel-toggle-update-time- stamp-on-save &optional GLOBALLY)	 Toggle time-stamp update on file save and display current state. By default change behaviour for local buffer only. When GLOBALLY argument is non-nil, using any prefix argument, change it for all buffers for the current Emacs editing session (the change does not persist across Emacs sessions). To modify the global state permanently modify the customized value of the 'pel-update-time-stamp' user option via the 'pel-pkg-for-filemng' group customize buffer with <f11> f <f2> 1.</f2></f11> This command is only available when the pel-update-time-stamp is set to t.
Toggle delete trailing space on save See also: Whitespace	• <f11> M-W • <f11> t w M-W</f11></f11>	(pel-toggle-delete-trailing- space-on-save &optional GLOBALLY)	Toggle deletion of trailing spaces on file save and display current state. • By default change behaviour for local buffer only. • When GLOBALLY argument is non-nil, using any prefix argument, change it for all buffers for the current Emacs editing session (the change does not persist across Emacs sessions). Trailing whitespace deletion is automatically activated on file save when the pel-delete-trailing-whitespace user-option is set to t. Use this command to de-activate it or re-activate it. • To modify the global state permanently modify the customized value of the 'pel-delete-trailing-whitespace' user option via the 'pel-pkg-for-filemng' group customize buffer with <f11> f <f2> 1.</f2></f11>

<u>Operation</u>	<u>Keystroke</u>	Function	<u>Note</u>	
Inserting & Automatically Updating Copyrights	Emacs has built-in support for insertion and update of copyright notices inside files. • Two commands, shown below, are provided to manually insert or update the file's copyright notice. • The copyright notice can be automatically updated by adding the copyright-update function to the list of before-save-hook variable with the following code: (add-hook 'before-save-hook 'copyright-update)			
<u>оорундико</u>		updated, the copyright notice ed as the first 2000 characters.	must be placed within an area at the beginning of the file specified by the value of the copyright-limit This variable is customizable.	
Insert copyright notice at point See also: <u>Notice Inserting</u> <u>Text</u>	<f11> i C</f11>	(copyright &optional STR ARG)	Insert a copyright by \$ORGANIZATION notice at cursor. • If the ORGANIZATION environment variable is not available, Emacs prompts for it.	
Update file's copyright notice	M-x copyright- update	(copyright-update &optional ARG INTERACTIVEP)	Update copyright notice to indicate the current year. • With prefix ARG, replace the years in the notice rather than adding the current year after them. If necessary, and 'copyright-current-gpl-version' is set, any copying permissions following the copyright are updated as well.	
			bes not warn if there is no copyright in the current buffer to update.	
	 It does not create a missing notice. If you want to be prompted automatically to update an existing but out-of-date copyright notice, write the following inside your init.el file: (add-hook 'before-save-hook 'copyright-update) 			
Automatic File Time Stamp on file save	Emacs has a built-in <u>automatic time-stamping of files.</u> It must be activated by adding the <u>time-stamp</u> function to the <u>before-save-hook</u> variable. This can either be done via Emacs customization system or explicitly inside your init file with the following code: (add-hook 'before-save-hook 'time-stamp) • The time stamp will be added to files that contain, inside their first 8 lines, a line that looks like one of the following: • Time-stamp: <>			
References: • <u>TimeStamps @</u> EmacsWiki		change these defaults and get E	Emacs to update all sorts of time stamp formats, even inside source code statements:	
Change time stamp format in: markdown file reStucturedText file	• time-stamp • time-si • time-si • time-si	-pattern consists of 4 parts, ea tamp-line-limit: identifies when tamp-start: identifies the text p tamp-end: identifies the end of		
See also: ∑Inserting Text	 time-stamp-format specifies the format of the time stamp. Something like "%:y-%02m-%02d %02H:%02S %u" to specify the date and time in ISO format, with the user login's name. time-stamp-time-zone specifies the time zone selection: nil: Emacs local time t: Universal time wall: system wall clock time TZ: controlled by a TZ environment variable The time-stamp-format and time-stamp-time-zone variables can be set in your init file or via the Emacs customization system. 			
	 They are defined in the time-stamp customization group. To change the format or the pattern preceding or after the automatically updated time stamp, it is best to use file local variables: this will allow automatic time stamp updates in files with various formats. As an example, see the top and end of the PEL manual raw format file. By default, the time-stamp string must be placed within the first 8 lines of the file, otherwise it will not be updated automatically. If you want it located somewhere else in your file set the time-stamp-lime-limit file local variable. 			
	pel-update-time- corresponding to can also toggle it To insert a non-update.	PEL provides the extra user-option to control the automatic generation of time-stamps: • pel-update-time-stamp user-option controls whether time-stamps are automatically update time stamps in all files where a valid time-stamp corresponding to Emacs settings as described above. Set it to t (the default) to allow automatic time stamp updates. Set it to nil to prevent them. Yo can also toggle it globally for the current editing session by using the <f11> f M-t key sequence. To insert a non-updatable time stamp, the PEL package provides a set of text insert commands which include inserting a time stamp. • See the S Inserting Text table for the appropriate commands.</f11>		
Update file time stamp	<f11> f t</f11>	(time-stamp)	Force update the time stamp string(s) in the current buffer. • Updates a time stamp of format recognized by <i>Emacs current settings</i> even when automatic time-stamp update is off. • More information about the " <i>Emacs current settings</i> " in the description block above.	
Toggle time stamp automatic update	<f11> f M-t</f11>	(time-stamp-toggle-active &optional ARG)	Toggle 'time-stamp-active', setting whether <f11> f t updates a buffer. • With ARG, turn time stamping on if and only if arg is positive.</f11>	
RFC-Mode	Browsing and reading F	RFC Files with the following rfc-	mode commands. PRequires ric-mode activated by pel-use-ric-mode,	
Read a specific RFC	<f11> B r</f11>	(rfc-mode-read NUMBER)	Read the RFC document NUMBER. Offer the number at point as default.	
Browse RFCs	<f11> B R</f11>	(rfc-mode-browse)	Browse through all RFC documents referenced in the index.	
<u>Directory Tree</u> <u>Browsers</u>	Emacs supports mechanisms to browse file directories. This includes: • Emacs built-in <u>Dired</u> directory editor, along with several extensions. You can have several different Dired buffers in an Emacs session. • The Emacs built-in <u>Speedbar</u> and its extensions. There can only be one instance of a Speedbar buffer and that can be inside another frame. • Several other external packages: <u>Neotree</u> , treemacs and <u>Ztree</u>			
View Directory Tree with NeoTree	The NeoTree external package provides a Vim-NerdTree like tree-view of a directory with expansion/collapse. PEL activates it when pel-use-neotree is set to t. • <f11> B N <f2> opens the PEL customization group to set pel-use-neotree.</f2></f11>			
	 <f11> B N <f3> prompts, select neotree to open the neotree customization group.</f3></f11> There is only one NeoTree window. It is a dedicated window. Icons used in the tree can be changed: In text mode set pel-neotree-font-in-terminal to arrows to use arrows instead of '+'. In graphics mode, if pel-neotree-font-in-graphics is set to icons then the icons provided by all-the-icons package is used. However, once PEL has installed the package it does not install the fonts. You must install the fonts manually by executing: M-x all-the-icons-install-fonts 			
View directory tree with NeoTree	<f11> B N N</f11>	(neotree-toggle)	Toggle show/hide the NeoTree window.	
	In the NeoTree buffer the following keys are available: • n next line, p previous line. • > end of buffer, < top buffer • SPC or RET or TAB: Open current item if it is a file, Fold/Unfold current item if it is a directory. • U Go up a directory • g Refresh • A Maximize/Minimize the NeoTree Window • It Toggle display hidden files. Controlled by neo-hidden-regexp-list user option. • O Recursively open a directory • C-c C-n Create a file or create a directory if filename ends with a '/' • C-c C-d Delete a file or a directory. • C-c C-c Change the root directory. • C-c C-p Copy a file or a directory.			
Open NeoTree for dir of current buffer	<f11> B N F</f11>	(neotree-find &optional PATH DEFAULT-PATH)	Open a NeoTree window using the directory of the current buffer. No prompt.	
Open NeoTree for specified directory	<f11> B N D</f11>	(neotree-dir PATH)	Prompt for a directory. Open a Neotree window for that directory.	
	I	1	6	

<u>Operation</u>	<u>Keystroke</u>	Function	<u>Note</u>	
Close NeoTree window	<f11> B N H</f11>	(neotree-hide)	Close the NeoTree window.	
Show NeoTree window	<f11> B N S</f11>	(neotree-show)	Show the NeoTree window.	
Treemacs • Manipulate directory trees associated as projects/workspaces • Manipulate the directories and files	The <u>treemacs</u> external package provides a workspace/project oriented tree-based view with expansion/collapse and actions of directories and files. PEL activates treemacs when the <u>pel-use-treemacs</u> user-option is turned on (set to t). Treemacs has a large number of user-options in the <u>treemacs</u> customization group and sub-groups. PEL <f11> B <f3> key sequence gives access to the customization group. On PEL, open (or close) the treemacs buffer with the <f11> B T key sequence.</f11></f3></f11>			
**	 In graphics mode the mouse provides access to most commands. In terminal (and graphics) mode when pain is inside the treemacs dedicated window, the treemacs major mode key-bindings, listed below, are available. 			
See: <u>∑X Treemacs</u>	The treemacs-mode an	d extensions have an extensive	e command set. See becommand.set">\sum x Treemacs for the complete list	
Open/close treemacs	<f11> B T</f11>	(treemacs)	Initialise or toggle treemacs. See **\sum x** Treemacs** for treemacs-mode commands. If the treemacs window is visible hide it. If a treemacs buffer exists, but is not visible show it. If no treemacs buffer exists for the current frame create and show it. If the workspace is empty additionally ask for the root path of the first project to add.	
View Directory Tree with ZTree	The <u>ztree</u> external package provides a text-based tree-view of a directory with expansion/collapse. PEL ztree customization: • <f11> B <f2> opens the PEL customization group (select the tree subgroup) . See also: Customize. • Modify one of the following PEL provided customization user options: • pel-ztree-dir-move-focus : set to t to move focus to new entry when <ret> is typed. • pel-ztree-dir-filter-list : add a list of regexp to ignore more file. Do not enter quote for string. For example, to ignore the .pyc files, enter ^.*pyc on a line. • pel-ztree-show-filtered-files : set to t to display filtered files until H is typed. Normally they are not shown until H is typed. • <f11> B <f3> prompts, select ztree to open the ztree customization group itself. 1. Execute M-x pel-init after settling and applying new values to activate the new values.</f3></f11></ret></f2></f11>			
View directory as tree with ztree-dir	<f11> B Z</f11>	(ztree-dir PATH)	Open an interactive buffer with the directory tree of the PATH given. Opens the tree buffer in the current window. There can be several buffers with different ztree-dir trees.	
		In the Ztree Dir buffer the following keys are available: > : narrow/display directory on current line		
Searching/Finding Files See also:	The following commands can be used to search for file by name or content. • See: Video: .Emacs #6: searching and finding files. Use man to get more information, • on locate: <f11> ? m locate • on find: <f11> ? m find</f11></f11>			
• <u>S Dired</u>	Source of the display of more than the file names.			
Search for file with locate	<f11> f L</f11>	(locate SEARCH-STRING &optional FILTER ARG)	Prompt for a search pattern and search for filenames using the system locate command line utility through the sell to search a database of all pathnames that match the specified search pattern. The database is recomputed periodically. • The search result is shown in a '*Locate*' buffer. • With prefix arg ARG, prompt for the exact shell command to run instead. This way you can specify options to the locate command line utility.	
		(counsel-locate &optional INITIAL-INPUT)	Call a "locate" style shell command with counsel listing and completion user-interface. • INITIAL-INPUT can be given as the initial minibuffer input. 1 This binding activated when the pel-use-counsel user-option is turned on. 2 When pel-use-ivy-hydra user-option is set you can activate the ivy-hydra with C-o. When Hydra is active, minibuffer editing is disabled and menus display short aliases: Short Normal Command name o C-g keyboard-escape-quit	
			j C-n ivy-next-line k C-p ivy-previous-line h M-< ivy-beginning-of-buffer l M-> ivy-end-of-buffer d C-m ivy-done f C-j ivy-alt-done g C-M-m ivy-call u C-c C-o ivy-occur	
Run grep via find	• <f11> f f g</f11>	(find-grep COMMAND-	Run grep via find, with user-specified args COMMAND-ARGS.	
See also: ∑ Grep	• <f11> g f</f11>	ARGS)	 Collect output in a buffer. While find runs asynchronously, you can use the C-x command to find the text that grep hits refer to. This command uses a special history list for its arguments, so you can easily repeat a find command. 	
Search for files with 'find' and open Dired buffer	<f11> f f d</f11>	(find-dired DIR ARGS)	Prompts for the root to search from, and a find command to search for files with the Unix find. • Specify the arguments for the <u>find command</u> . • For example, to perform a case insensitive search for all .h files, use: -iname "*\.h" • Opens a Dired-mode buffer and show the files found in there.	
Search directory for files and open Dired buffer for those	<f11> f f n</f11>	(find-name-dired DIR PATTERN)	Search DIR recursively for files matching the globbing pattern PATTERN, and run Dired on those files. • PATTERN is a shell wildcard (not an Emacs regexp) and need not be quoted. • The default command run (after changing into DIR) is: findname 'PATTERN' -1s	
Find files in a directory and open Dired output	<f11> f f h</f11>	(find-grep-dired DIR REGEXP)	Find files in DIR that contain matches for REGEXP and start Dired on output. The command run (after changing into DIR) is: find . \(-type f -exec 'grep-program' 'find-grep-options' -e REGEXP {} \; \) -ls where the first string in the value of the variable 'find-ls-option' specifies what to use in place of "-ls" as the final argument.	
Find Emacs Lisp files in	<f11> f f 1</f11>	(find-lisp-find-dired DIR	Find Emacs Lisp files in DIR, matching REGEXP.	
directory tree		REGEXP)	Open *Find Lisp Dired* buffer on output.	

File Management — References

Topic & Link	Description		
Emacs Display - Mode Line	Read first. Describes what the Emacs mode line displays.		
GNU Emacs Manual - File Handling	Describes how to open and deal with files and directories in Emacs.		
GNU EMACS Manual - Interactive Do	Describes the ido-mode, a nice addition that helps with completing file names at prompts.		
Display path of file in status bar	In graphics mode, display the buffer name and the full path file in parenthesis inside the frame title bar.		
How do I rename an open file in Emacs?			
Find files faster with the recent files package	Mickey Petersen article describing the recent file feature. PEL ido-recentf-open is taken from Mickey Peterson code.		