
















Projectile - Project Interaction

| Description | Keystroke | Function | Note |
|--|--|---|---|
| Projectile | <div>  The projectile external package is a great package to manage your project. See the projectile user manual for more information. </div> <div>  PEL activates projectile when the pel-use-projectile user option is non-nil. If it is set to t projectile is available but you must explicitly activate it with the projectile-mode command (which PEL binds to <f11> <f8>) Then PEL binds projectile-command-map to <f8> instead of projectile recommended binding C-c p. </div> <div> If you want projectile-mode available when Emacs starts, then set the user option to use-from-start instead. </div> | | |
| Open local help PDF | <div> <ul style="list-style-type: none"> <f11> <f8> <f1> <f8> <f1> </div> | (pel-help-pdf &optional OPEN-WEB-PAGE) | Open the Projectile local PDF. If the prefix argument (like C-u or M--) is used, then it opens the remote GitHub hosted raw PDF instead. If the pel-flip-help-pdf-arg user-option is set it's the other way around. <ul style="list-style-type: none"> The key sequence <f8> <f1> is only available when the projectile mode is activated. |
| » Customize PEL support for Projectile | <div> <ul style="list-style-type: none"> <f11> <f2> P <f8> <f11> <f8> <f2> <f8> <f2> </div> | (pel-cfg-pkg-project-mng &optional OTHER-WINDOW) | Customize PEL Project-Mng support. <ul style="list-style-type: none"> If OTHER-WINDOW is non-nil (use C-u), display in other window and open the related group(s) that exist. The key sequence <f8> <f2> is available when the projectile mode is activated. |
| » Customize Projectile | <div> <ul style="list-style-type: none"> <f11> <f8> <f3> <f8> <f3> </div> | (pel-customize-projectile) | Open the projectile customization group where you can modify projectiles configuration. <ul style="list-style-type: none"> The key sequence <f8> <f3> is available when the projectile mode is activated. |
| | | | |
| Toggle projectile mode | <f11> <f8> <f8> | (projectile-mode &optional ARG) | Toggle projectile-mode, a minor mode to assist project management and navigation. <div>  PEL activates the projectile-mode when Emacs starts if the pel-use-projectile user option is set to use-from-start. If instead pel-use-projectile is set to t, then you must use this command to activate it. </div> |
| Projectile Commander Invoke other Projectile commands with single letters | <f8> m | (projectile-commander) | Execute a Projectile command with a single letter. <ul style="list-style-type: none"> The user is prompted for a single character indicating the action to invoke. <ul style="list-style-type: none"> The '?' character describes then available actions. You can define new <i>methods</i> by writing Emacs Lisp code which uses the def-projectile-commander-method to assign specific keys in the map for the action expressed in Emacs Lisp code. <ul style="list-style-type: none">  Currently PEL does not provide a mechanism to define this type of code, but you can add this code inside your init.el file after the call to pel-init(). |
| Define Projects | Projectile identifies project files with the information it gets from VCS repo definition files (such as .git or .hg) or from the presence and content of a .projectile file in the root directory of a project. Read the Projectile//Docs/Project for more information. | | |
| Configure the project | <f8> C | (projectile-configure-project ARG) | Run project configure command. <ul style="list-style-type: none"> Normally you'll be prompted for a compilation command, unless variable 'compilation-read-command'. You can force the prompt with a prefix ARG. |
| Edit/Create a .dir-locals.el file for the project | <f8> E | (projectile-edit-dir-locals) | Edit or create a .dir-locals.el file of the current project. <ul style="list-style-type: none"> The file is located at the root directory of the project. Command prompts for names of variables and their values and writes the corresponding Emacs Lisp code inside the file. |
| Navigate to projects | Projectile learn about projects when you first visit (or open) a file that is in a repo directory tree or a directory tree that has a .projectile file at its root. See the Projectile Supported Project Types for the list of project types supported and how to add support for more. You can change the active project with the following commands and potentially execute commands on them. | | |
| Switch project | <f8> p | (projectile-switch-project &optional ARG) | Switch to a project previously visited. we have visited before: <ul style="list-style-type: none"> Prompt for the name of the project, showing the previously visited projects. Invokes the command referenced by 'projectile-switch-project-action' on switch. <ul style="list-style-type: none"> By default this is projectile-find-file: it prompts for a file within the project With a prefix ARG invokes 'projectile-commander' instead of 'projectile-switch-project-action.' This allows running something else not eh other project. |
| Switch to a currently opened project | <f8> q | (projectile-switch-open-project &optional ARG) | Switch to a project we have currently opened. <ul style="list-style-type: none"> Invokes the command referenced by 'projectile-switch-project-action' on switch. With a prefix ARG invokes 'projectile-commander' instead of 'projectile-switch-project-action.' |
| Expand Speedbar to the entire projectile project directories and files See also: » Speedbar | <f8> M-s | (projectile-speedbar-open-current-buffer-in-tree) | With a speedbar already opened, expand it to the entire current projectile project directories and files. <div>  Requires the projectile-speedbar external project. </div> <div>  PEL activates it when the pel-use-projectile-speedbar user-option is set to t. </div> <div>  To use this command you must first activate projectile. </div> <div>  The command fails if issued when point lies inside the speedbar window. </div> |
| Protect project files | | | |
| Toggle project read-only | <f8> ~ | (projectile-toggle-project-read-only) | Toggle project read only. |
| Project Buffers See also: » Buffers | Projectile provides the following commands to manage buffers related to a project. <ul style="list-style-type: none"> projectile-buffer opens a Buffer listing only the files in the currently active project, nothing else. Then shown is a set of commands open a buffer that is part of a project, in the current window, another window with and without selecting it, or into another frame. <ul style="list-style-type: none"> When prompting for a buffer name, the lists of available buffers is restricted to the buffers that are part of the current project (as opposed to all buffers currently opened in Emacs). The last 2 commands listed are: <ul style="list-style-type: none"> save all buffers related to the project, kill all buffers related to the project. | | |
| List project's buffers | <f8> I | (projectile-ibuffer PROMPT-FOR-PROJECT) | Open an IBuffer window showing all buffers in the current project (exclude all others). <ul style="list-style-type: none"> Let user choose another project when PROMPT-FOR-PROJECT is supplied. |
| Switch to previous project buffer | <f8> <left> | (projectile-previous-project-buffer) | In selected window switch to the previous project buffer. <ul style="list-style-type: none"> If the current buffer does not belong to a project, call 'previous-buffer'. |
| Switch to next project buffer | <f8> <right> | (projectile-next-project-buffer) | In selected window switch to the next project buffer. <ul style="list-style-type: none"> If the current buffer does not belong to a project, call 'next-buffer'. |
| Switch to a project buffer | <f8> b | (projectile-switch-to-buffer) | Switch to a specific project buffer: prompt for the buffer name. |
| Switch to a project buffer in other window | <f8> 4 b | (projectile-switch-to-buffer-other-window) | Switch to a specific project buffer and show it in another window. |
| Open a project buffer in other window with our selecting it. | <f8> 4 C-o | (projectile-display-buffer) | Display a project buffer in another window without selecting it. |
| Switch to a project buffer in other frame | <f8> 5 b | (projectile-switch-to-buffer-other-frame) | Switch to a project buffer and show it in another frame. |
| Save all project buffers | <f8> S | (projectile-save-project-buffers) | Save all project buffers. |
| Kill all project's buffers | <f8> k | (projectile-kill-buffers) | Kill project buffers. Prompts for confirmation before killing the buffers. |

| Description | Keystroke | Function | Note |
|---|---|---|--|
| Search in Project <ul style="list-style-type: none"> Searching in project buffers: projectile provides the multi-occur for project buffers, shown non the first row. Searching in project files: projectile provides the following recursive-grep like search tools, they are listed starting on the second row. <ul style="list-style-type: none"> The first one searches inside buffers, not in files. That may be useful when looking for unsaved buffers or for special buffers. The last 2 require external packages and external command line utilities that must have been installed separately: ripgrep and ag. The ripgrep and ag searches are faster than the standard grep search. To navigate through search results use: <ul style="list-style-type: none"> move to next occurrence: C-x ` or M-g n or M-g M-n move to previous occurrence: M-g p or M-g M-p | | | |
| Search for occurrence of text in project buffers | <f8> o | (projectile-multi-occur &optional NLINES) | Do a ‘multi-occur’ in the project’s buffers . <ul style="list-style-type: none"> With a prefix argument, show NLINES of context. |
| Search in project files with recursive grep | <f8> s g | (projectile-grep &optional REGEXP ARG) | Perform rgrep in the project. <ul style="list-style-type: none"> With a prefix ARG asks for files (globbing-aware) which to grep in. With prefix ARG of ‘-’ (such as ‘M--’), default the files (without prompt), to ‘projectile-grep-default-files’. With REGEXP given, don’t query the user for a regexp. |
| Search in project files with ripgrep <ul style="list-style-type: none"> Rust (ripgrep) regexp syntax | <f8> s r | (projectile-ripgrep SEARCH-TERM &optional ARG) | Run a Ripgrep search with ‘SEARCH-TERM’ at current project root. <ul style="list-style-type: none"> With an optional prefix argument ARG SEARCH-TERM is interpreted as a regular expression.  Requires the projectile , ripgrep.el external packages as well as the ripgrep command line utility.  PEL activates this command when pel-use-projectile is non-nil. But to make it work you must also set pel-use-ripgrep to t . Also note that the ripgrep command line utility must be installed manually. |
| Search in project files with ag <ul style="list-style-type: none"> PCRE regex syntax | <f8> s s | (projectile-ag SEARCH-TERM &optional ARG) | Run an ag search with SEARCH-TERM in the project. <ul style="list-style-type: none"> With an optional prefix argument ARG SEARCH-TERM is interpreted as a regular expression.  Requires the projectile , ag.el external packages as well as the ag command line utility.  PEL activates this command when pel-use-projectile is non-nil. But to make it work you must also set pel-use-ag to t . Also note that the ag command line utility must be installed manually. |
| Replace in Project | Text replacement inside all project files. | | |
| Replace test in project files | <f8> r | (projectile-replace &optional ARG) | Replace literal string in project using non-regexp ‘tags-query-replace’. <ul style="list-style-type: none"> With a prefix argument ARG prompts you for a directory on which to run the replacement. |
| Project CTags  <ul style="list-style-type: none"> generate TAGS file jump to definition See Xref for commands using the generated tags. | Projectile supports the ctags file cross referencing utility by being able to regenerate the tags/TAGS file explicitly or periodically via a timer. <ul style="list-style-type: none"> The tool used is controlled by the projectile-tags-backend user option. The default is set to auto. <ul style="list-style-type: none"> The shell command projectile uses is identified in the projectile-tags-command user option. The default is set to <code>ctags -Re -f "%s" %s "%s"</code>. <ul style="list-style-type: none"> This is the command line for the now defunct exuberant-ctags. That project is no longer maintained. You can use Universal Ctags instead, it is also called ctags. That is unfortunate because that conflicts with GNU ctags and Emacs ctags. One way to solve the problem is to create links to the Universal Ctags executable named uctags and do the same for readtags (ureadtags), then update the value of projectile-tags-command to <code>uctags -Re -f "%s" %s "%s"</code>. Another solution is to rename other of those programs and leave ctags for the Universal ctags. You can also set projectile-tags-backend to use gtags: the GNU global tagging system. If you want projectile to automatically regenerate the tags periodically, customize the projectile-enable-idle-timer user option. PEL provides the <f8> <f1> key binding to access the projectile customization group quickly. | | |
| Regenerate project's Tags file | <f8> R | (projectile-regenerate-tags) | Regenerate the project’s [e]gtags. |
| Jump to project's (tag) symbol definition | <f8> j | (projectile-find-tag) | Find tag in project. <ul style="list-style-type: none"> Prompt for a target (like the name of a function) and then open the file at that location. Tag defaults to symbol at point. To go back where point was, use M- , |
| Run Projects commands | | | |
| Run project compilation command | <f8> c | (projectile-compile-project ARG) | Run project compilation command. <ul style="list-style-type: none"> Normally you’ll be prompted for a compilation command, unless variable ‘compilation-read-command’. You can force the prompt with a prefix ARG. The result of the command is shown in a *compilation* buffer that supports the following keys: <ul style="list-style-type: none"> TAB : move to next error backtab : move to previous error C-o : move to the source where the error is reported o : open a visible link in a compilation-mode g : re-run the command |
| Execute project's “run” command | <f8> u | (projectile-run-project ARG) | Run project run command. <ul style="list-style-type: none"> Normally you’ll be prompted for a compilation command, unless variable ‘compilation-read-command’. You can force the prompt with a prefix ARG. The result is also shown in a *compilation* buffer, with the same key commands available. |
| Project & VCS | Projectile provides the following commands to interoperate with your project VCS. | | |
| Browse dirty version controlled projects | <f8> v | (projectile-browse-dirty-projects &optional CACHED) | Browse dirty version controlled projects. <ul style="list-style-type: none"> With a prefix argument, or if CACHED is non-nil, try to use the cached dirty project list.  Use this to quickly identify your projects that have non-committed files.  This may take some time to execute. |
| Open the project VCS status buffer | <f8> v | (projectile-vc &optional PROJECT-ROOT) | Open ‘vc-dir’ at the root of the project. <ul style="list-style-type: none"> For git projects ‘magit-status-internal’ is used if available. For hg projects ‘monky-status’ is used if available. If PROJECT-ROOT is given, it is opened instead of the project root directory of the current buffer file. If interactively called with a prefix argument, the user is prompted for a project directory to open. |
| Project's Tests | | | |
| Toggle between and implementation file and its test file | <f8> t | (projectile-toggle-between-implementation-and-test) | Toggle between an implementation file and its test file. |
| Open matching test file in other window | <f8> 4 t | (projectile-find-implementation-or-test-other-window) | Open matching implementation or test file in other window. |
| Open matching implementation file in other frame | <f8> 5 t | (projectile-find-implementation-or-test-other-frame) | Open matching implementation or test file in other frame. |
| Run project's test | <f8> p | (projectile-test-project ARG) | Run project test command. <ul style="list-style-type: none"> Normally you’ll be prompted for a compilation command, unless variable ‘compilation-read-command’. You can force the prompt with a prefix ARG. |

| Description | Keystroke | Function | Note |
|--------------------------------------|---|---|---|
| Jump to project test file | <f8> T | (projectile-find-test-file &optional INVALIDATE-CACHE) | Jump to a project's test file using completion. <ul style="list-style-type: none"> With a prefix arg INVALIDATE-CACHE invalidates the cache first. |
| Run Shell Commands in Project's root | The following projectile commands execute shell commands in the project's root directory. | | |
| Run shell command in project root | <f8> ! | (projectile-run-shell-command-in-root) | Invoke 'shell-command' in the project's root. <ul style="list-style-type: none"> Display results in a *Shell Command Output* buffer. |
| Run async command in project root | <f8> & | (projectile-run-async-shell-command-in-root) | Invoke 'async-shell-command' in the project's root. <ul style="list-style-type: none"> Display results in a *Shell Command Output* buffer. |
| Specialized Shells | Open a specialized shell related to the current project. The buffer name reflect the name of the project's root directory. See the Σ Shells table for more information on these shells. | | |
| eshell | <f8> x e | (projectile-run-eshell ARG) | Invoke 'eshell' in the project's root. <ul style="list-style-type: none"> Switch to the project specific eshell buffer if it already exists. Use a prefix argument ARG to indicate creation of a new process instead. |
| gdb | <f8> x g | (projectile-run-gdb) | Invoke 'gdb' in the project's root. |
| ielm | <f8> x i | (projectile-run-ielm ARG) | Invoke 'ielm' in the project's root. <ul style="list-style-type: none"> Switch to the project specific ielm buffer if it already exists. Use a prefix argument ARG to indicate creation of a new process instead. |
| shell | <f8> x s | (projectile-run-shell ARG) | Invoke 'shell' in the project's root. <ul style="list-style-type: none"> Switch to the project specific shell buffer if it already exists. Use a prefix argument ARG to indicate creation of a new process instead. |
| term | <f8> x t | (projectile-run-term ARG) | Invoke 'term' in the project's root. <ul style="list-style-type: none"> Switch to the project specific term buffer if it already exists. Use a prefix argument ARG to indicate creation of a new process instead. |
| vterm | <f8> x v | (projectile-run-vterm &optional ARG) | Invoke 'vterm' in the project's root. <ul style="list-style-type: none"> Switch to the project specific term buffer if it already exists. Use a prefix argument ARG to indicate creation of a new process instead. |

Projectile — References

| Topic & Link | | Note |
|---|--|--|
| Projectile | | |
| Projectile @ Github | | |
| Projectile User Manual | | |
| | | |
| Recommended tools | | When using Projectile it is best to also install these command line tools to get the best performance. |
| Find Replacements | fd | fd is a file finder hat performs much faster than find. |
| Grep Replacements | ripgrep : rg | ripgrep is a fast grep. The command line name is rg. Uses Rust regular expressions . Read the author's blog on ripgrep for more info. <ul style="list-style-type: none"> Note: ripgrep and ag both support .ignore file to identify the file patterns to ignore in its search. See this blog entry for history. |
| | ag | ag is another fast grep. Uses PCRE regular expressions. |
| Ctags Cross Reference tools | Universal Ctags | A ctag program, successor of the now defunct Exuberant CTags . See Universal CTags Github repository and the Universal CTags Hacking Guide for more info. |
| | Gnu Global | |
| | | |