Helm User Manual

ThierryVolpiatto

Last update : 24 novembre 2012

Table des matières

1 Install

First get the files from git repo :
Helm git repo is at :
https://github.com/emacs-helm/helm
You will find there tarballs of differents versions.

To get it with git :

```
git clone https://github.com/emacs-helm/helm
```

NOTE : Files are published on Emacswiki, but be aware that it is unsafe to get files from Emacswiki,
thus, helm is not maintained anymore on Emacswiki, so files found there should be deprecated.

Once you have the helm directory, 'cd' to it and run 'make'.
Add it now to your 'load-path' :

```
(add-to-list 'load-path "/path/to/helm/directory")
```

Manual installation :

You need 3 files :
– 1) helm.el
Contain the helm engine.
– 2) helm-config.el
Contain all the sources and preconfigured functions ready to use.
– 3) helm-match-plugin.el
Allow matching multi pattern search when entering a space in prompt.
Once downloaded these files, put them in your 'load-path' and byte-compile them. If you don't know what is your load-path do C-h v load-path.

2 Config

Edit your /.emacs.el file and add :

```
(require 'helm-config)
```

NOTE : It is not recommended to use the variable 'helm-sources', please use instead the preconfigured helm command you will find in helm-config.el or build your own.
Be aware also that making your own helm commands with a lot of sources involved can be very costly and slowdown helm a lot.

3 General helm commands

Helm allow you to have few binding to remember unlike all others Emacs applications.
Thus, all bindings are auto documented.

Helm show you by default in mode-line the most useful bindings, you will see in headers of helm buffer some more specific commands.

So when helm start what you have to remember :

- Access to action menu with
  `TAB`
- Use persistent actions with
  `C-z`
- Mark candidate with
  `M-<SPACE>`

So three bindings to remember and they are anyway documented in mode-line. For more, hitting

```
C-h m
```

while in helm session will show you all other bindings.
NOTE : Some helm commands have a special keymap, you can access infos on these keymap with 'C-c ?', it should be documented in mode-line.

4 Overview of preconfigured helm commands

For starting with helm, a set of commands have been set for you in helm menu. The bindings of all these commands are prefixed with 'f5-a'.

To discover more helm commands run from menu helm all commands (helm-execute-helm-command). Or run helm-M-x (f5-a M-x) and type helm.

When you like a command, e.g f5 a M-x you should bind it to something more convenient like M-x to replace the Emacs original keybinding.

5 Helm Find Files

'helm-find-files' provide you a way to navigate in your system file easily. All the actions you can do on files from here are described in this section.

It is binded in menu, and in 'helm-command-map' to f5-a C-x C-f.
We will assume you have binded 'helm-find-files' to C-x C-f. To do that put in your .emacs.el :

```
(global-set-key (kbd "C-x C-f") 'helm-find-files)
```

It is well integrated with tramp, you can enter any tramp filename and it will complete. (e.g /su : :, /sudo : :, /ssh :host :, ... etc)

Called with a prefix arg, (C-u) helm-find-files will show you also history of last visited directories.

5.1 Navigation

Helm-find-files is not by default on / but on default-directory or thing-at-point as it use ffap.If you are on a url, a mail adress etc.. it will do the right thing.

So helm-find-files work like find-file (C-x C-f), but if you use it with helm-match-plugin.el, you have to add a space and then the next part of pattern you want to match :

Example :

```
Find Files or url: ~/
That show all ~/ directory.

Find Files or url: ~/des
will show all what begin with "des"

Find Files or url: ~/ esk
(Notice the space after ~/) will show all what contain esk.

Find Files or url: ~/ el$
Will show all what finish with el
```

You can move in the helm buffer with C-n C-p or arrow keys, when you are on a file, you can hit C-z to show only this file-name in the helm buffer. On a directory, C-z will switch to this directory to continue searching in it. On a symlink C-z will expand to the true name of symlink.(moving your mouse cursor over a symlink will show the true name of it).

So it is quite easy to navigate in your files with helm-find-files.

Forget to mention C-. that go to root of current dir or to precedent level of dir. So for example you can hit C-z and then come back immediatly where you were with C-. instead of erasing minibuffer input with DEL. On non graphic display, it is bound to C-l.
If 'helm-ff-lynx-style-map' is non–nil, you will be able to use 'left' instead of C-l and 'right' instead of C-z.

If you like it, you can safely bind it to C-x C-f to replace the standard find-file :

```
(global-set-key (kbd "C-x C-f") 'helm-find-files)
```

NOTE : Starting helm-find-files with C-u will show you a little history of the last visited directories.

5.2 Jump with nth commands

Take advantage of the second, third and 4th actions in helm. Instead of opening action menu with TAB, just hit :

C-e for 2th action
C-j for 3th action

You can bind 4th action to some key like this :

```
(define-key helm-map (kbd "<C-tab>") 'helm-select-4th-action)
```

5.3 Helm find files action shortcuts

Instead of having to open action pannel with TAB, you have some convenients shortcuts to quickly run actions. Use C-c ? from an helm-find-files session to have a description.

Command	Key
helm-ff-run-grep	M-g s (C-u recurse)
helm-ff-run-rename-file	M-R (C-u Follow)
helm-ff-run-copy-file	M-C (C-u Follow)
helm-ff-run-byte-compile-file	M-B (C-u Load)
helm-ff-run-load-file	M-L
helm-ff-run-symlink-file	M-S (C-u Follow)
helm-ff-run-delete-file	M-D
helm-ff-run-switch-to-eshell	M-e
helm-ff-run-complete-fn-at-point	M-tab
helm-ff-run-switch-other-window	C-o
helm-ff-run-open-file-externally	C-c C-x (C-u choose)
helm-ff-help	C-c ?
helm-ff-rotate-left-persistent	M-l
helm-ff-rotate-right-persistent	M-r
helm-find-files-down-one-level	C-. or C-l
helm-ff-properties-persistent	M-i

5.4 Turn in image viewer

You can turn helm-find-files in a nice image-viewer.

Navigate to your image directory, then type C-u C-z on first image. Now turn on 'follow-mode' with C-c C-f. You can now navigate in your image directory with arrow up and down or C-n C-p. Don't forget also to use C-t to split you windows vertically if needed.

You will find also two actions to rotate image in action menu. To use these actions whitout quitting, use M-l (rotate left) and M-r (rotate right). Of course M-l and M-r have no effect if candidate is not an image file.

Don't forget to use 'C-t' to split windows vertically, and then

`'C-}' and 'C-{'`

to narrow/enlarge helm window.

NOTE : It use image-dired in background, so if image-dired don't work for some reason, this will not work too. Be sure to have Imagemagick package installed.

5.5 Serial rename

You can rename files with a new prefix name and by incremental number. The marked files will be renamed with a new prefix name and starting at the start-number you have choosen.
Note that the marked files are in the order of the selection you did, this allow to reorder files.
If you mark files in other directories than the current one, these files will be moved or symlinked to current one.

TIP : If you have more than 100 files to serial rename, start at 100 instead of one to have your directory sorted correctly.

You have to ways to serial rename :

- By renaming : All the file of others directories are moved in directory where renaming happen.

- By symlinking : All the files that are not files of the directory where you want to rename will be symlinked, others will be renamed.

- By copying : All the file of others directories are copied in directory where renaming happen.

Example of Use :
I want to create a directory with many symlinked images coming from various directories.
1) C-x C-f (launch helm-find-files)
2) Navigate to the place of your choice and write new directory name ending with "/" and press RET.
3) Navigate and browse images, when you want an image mark it, you can mark all in a directory with M-a.
4) When you have marked all files, choose "serial rename by symlinking" in action menu.
5) Choose new name and start number.
6) Navigate to initial directory (where files will be renamed/symlinked) and RET.
7) Say yes to confirm, that's done.
8) Start viewing your pictures.

5.6 Grep

We describe here helm-do-grep, an incremental grep. It is really convenient as you can start a search just after finding the place or file(s) you want to search in. By the nature of incremental stuff, it is faster than original Emacs grep for searching.

As you type the display change (like in all other helm commands). This grep is also recursive unlike the emacs implementation that use find/xargs.

It support wildcard and (re)use the variables 'grep-find-ignored-files' and 'grep-find-ignored-directories'.

It have full tramp integration. (you can grep file on a remote host or in su/sudo methods).

- NOTE : You will find a file named helm-grep.el in extensions. It is DEPRECATED and NOT needed to use with what is described here. It is another implementation of grep but not incremental.
- NOTE : When using it recursively, 'grep-find-ignored-files' is not used unless you don't specify the only extensions of files where you want to search (you will have a prompt). You can now specify more than one extension to search.
  e.g *.el *.py *.tex
  will search only in files with these extensions.
- NOTE : Windows users need grep version $\geq$ 2.5.4 of Gnuwin32 on windoze. This version should accept the –exclude-dir option.

Grep in action on current file :

5.7 Helm do grep

Start with M-x helm-do-grep bound to f5 a M-g s A prefix arg will launch recursive grep.

5.8 Grep from helm-find-files

From helm-find-files (f5 a C-x C-f) Open the action menu with tab and choose grep. A prefix arg will launch recursive grep.

- NOTE :You can now launch grep with (C-u) M-g s without switching to the action pannel.

5.9 Grep One file

Just launch grep, it will search in file at point. if file is a directory, it will search in ALL files of this directory like :

```
grep -nH -e pattern *
```

5.10 Grep Marked files

Just mark some files with

```
C-<SPACE>
```

and launch grep.

5.10.1 Grep marked files from differents directories

This is a very nice feature of helm grep implementation that allow to search in specific files located not only in current directory but anywhere in your file system.

To use navigate in your file system and mark files with

```
C-<SPACE>
```

When you have marked all files, just launch grep in action menu.

NOTE : Using prefix-arg (C-u) will start a recursive search with the extensions of the marked files except if those are one of “grep-find-ignored-files”.

5.11 Grep Directory recursively

From helm-find-files, reach the root of the directory where you want to search in, then hit TAB to open the action menu and choose grep with a prefix arg (i.e C-u RET).

if you want to use grep directly from helm-do-grep, do :

```
C-u f5 a M-g s
```

You will be prompted for selecting in which category of files to search : Use the wilcard syntax like *.el for example (search in only ".el" files).

- NOTE : Be sure to be at the root of directory, of course grepping recursively with cursor on a filename candidate will find nothing.

5.12 Grep Using Wildcard

You can use wildcard : From the root of your directory, if you want for example to search files with .el extension : add *.el to prompt.

5.13 Grep thing at point

Before lauching helm, put your cursor on the start of symbol or sexp you will want to grep. Then launch helm-do-grep or helm-find-files, and when in the grep prompt hit C-w as many time as needed.

5.14 Grep persistent action

As always, C-z will bring you in the buffer corresponding to the file you are grepping.
Well nothing new, but using C-u C-z will record this place in the mark-ring. So if you want to come back later to these places no need to grep again, you will find all these places in the mark-ring.
Accessing the mark-ring in Emacs is really inconvenient, fortunately, you will find in helm-config "helm-all-mark-ring" which is a mark-ring browser (f5-a C-c SPACE).

"helm-all-mark-ring" is in helm menu also, in the tool section.

- TIP : Bind "helm-all-mark-ring" to C-c SPACE.
  `(global-set-key (kbd "C-c <SPC>") 'helm-all-mark-rings)`
- NOTE : "helm-all-mark-ring" handle global-mark-ring also.

5.15 Save grep session

If you want to save the results of your grep session, doing "C-x C-s" will save your grep results in a grep-mode buffer.

5.16 Open Files Externally

You will find in action menu from helm-find-files an action to open file with external program. If you have no entry in .mailcap or /etc/mailcap, you will enter an helm session to choose a program to use with this kind of file. It will offer to you to save setting to always open this kind of files with this program. Once configured, you can however open the files of same extension with some other program by forcing helm to choose program with C-u.

NOTE : You can now open files externally with "C-c C-x" from helm-find-files.

5.17 Eshell command on files

You can run eshell-command on files or marked files, the command you use have to accept one file as argument. The completion is make on your eshell aliases. This allow you creating personal actions for "helm-find-files".

5.18 Why Eshell

- Because eshell allow you to create aliases.
- Because eshell accept shell commands but also elisp functions.

All these command should end with $1. You will have completion against all these aliases once eshell is loaded. (start it once before using helm-find-files).

5.19 Setup Aliases

Go in eshell, an enter at prompt :

```
alias my_alias command \$1
```

NOTE : don't forget to escape the $.
See the documentation of Eshell for more info.

5.20 Problem starting Eshell

Eshell code is available (autoloaded) only when you have started once eshell. That's annoying like many autoloaded stuff in Emacs.

Here how to start Eshell at emacs startup :

Add this to your .emacs :

```
(add-hook 'emacs-startup-hook #'(lambda ()
                                  (let ((default-directory (getenv "HOME")))
                                    (command-execute 'eshell)
                                    (bury-buffer))))
```

5.21 Dired Commands

To enable some of the usual commands of dired, put in .emacs.el

```
(helm-dired-bindings 1)
```

Or run interactively :

```
M-x helm-dired-bindings
```

This will replace in dired C, R, S, and H commands. That is copy, rename, symlink, hardlink. When creating a symlink, you will find relsymlink in actions menu.(TAB).
NOTE : This is deprecated for Emacs24 users, use instead 'ac-mode'.

5.22 Copy Files

It is a powerfull feature of helm-find-files as you can mark files in very different places in your file system and copy them in one place.

Dired is not able to do that, you can mark files only in current dired display and copy them somewhere.

So, easy to use, just mark some files, and hit copy files in the action menu. That will open a new helm-find-files where you can choose destination.

5.23 Rename Files

Just mark some files, and hit rename files in the action menu. That will open a new helm-find-files where you can choose destination.

5.24 Symlink Files

Just mark some files, and hit symlink or relsymlink files in the action menu. That will open a new helm-find-files where you can choose destination.

5.25 Hardlink

Just mark some files, and hit hardlink files in the action menu. That will open a new helm-find-files where you can choose destination.

5.26 Follow file after action

A prefix arg on any of the action above, copy, rename, symlink, hardlink, will allow you to follow the file. For example, when you want to copy an elisp file somewhere and you want to compile it in this place, hitting C-u RET will bring you in this place with the file already marked, you have just to go in action menu to compile it.

5.27 In Buffer File Completion

In any buffer and even in minibuffer if you have enable recursive-minibuffer

```
(setq enable-recursive-minibuffers t)
```

You can have completion with C-x C-f and then hit tab to choose action Complete at point
once you have found the filename you want.

5.28 Create File

Navigate to the directory where you want to create your new file, then continue typing the name of your file and type enter.
NOTE : If your filename ends with a / you will be prompt to create a new directory.

5.29 Create Directory

Navigate to the directory where you want to create your new directory, then continue typing the name of new directory - Parents accepted -
and end it with / type enter, you will be prompt to create your new directory (possibly with parents).

5.30 Ediff files

Well, that is easy to use, move cursor to a file, hit ediff in action menu, you will jump in another helm-find-files to choose second file.

5.31 Ediff merge files

move cursor to a file, hit ediff merge in action menu, you will jump in another helm-find-files to choose second file.

5.32 Browse archive with avfs

If you have installed avfs (See : http ://sourceforge.net/projects/avf) you can browse archives in your directory .avfs once it is mounted with "mountavfs".

Just move on the archive filename and press C-z (persistent action) and you will see in helm buffer the subdirectories of archive, just navigate inside as usual.

5.33 Display with icons

You can have a more fancy display showing icons for files and directories. Just add in .emacs :

```
(setq helm-c-find-files-show-icons t)
```

NOTE : This will slowdown helm-find-files unless you have a fast computer.

6 Helm write buffer

That is a replacement of standard write-buffer Emacs command with helm completion.

7 Helm insert file

That is a replacement of standard insert-file Emacs command with helm completion.

8 Helm M-x

It is binded to f5 a M-x, you should bind it to M-x.
Features :

- You can use prefix arguments before or during M-x session
- C-z is a toggle documentation for this command
- The key binding of command are shown.

9 Helm regexp

This is a replacement of regexp-builder. The groups are shown in a convenient way.

9.1 Query replace regexp

Write your regexp in helm-regexp, when it match what you want, you can run query-replace from action menu. NOTE : Before running helm-regexp, you can select a region to work in, that will narrow this region automatically.

9.2 Save regexp as sexp

When you use this, it will save your regexp for further use in lisp code, with backslash duplicated.

9.3 Save regexp as string

Save the regexp as you wrote it.

10 Helm locate

First be sure you have a locate program installed on your system. Most GNU/Linux distro come with locate included, you update or create the data base with "updatedb" command.

10.1 Search files

To use, just launch

```
M-x helm-locate (f5 a l)
```

Then enter filename at prompt. It will search this pattern entered also in directory and subdirectory names, to limit your search to basename, add "-b" after pattern. The search is performed on all files known in database, they maybe not exists anymore, so to limit to really existing files add after pattern "-e". To limit you search to specific number of results, use "-n" after your pattern with the number of results you want.

Example :

```
Pattern: emacs -b -e -n 12
```

10.2 Launch grep

When search is done, you can search in a specific file or directory with grep that you will find in action menu (TAB).

- NOTE :You can now launch grep with (C-u) M-g s without switching to the action pannel.

10.3 Use a local locate DB

You can specify a specific database with prefix argument ARG (C-u). Many databases can be used : navigate and mark them. See also 'helm-locate-with-db'.

To create a user specific db, use :

```
updatedb -l 0 -o dbpath -U directory
```

Where "dbpath" is a filename matched by "helm-locate-db-file-regexp"

10.4 Windows specificity

On Windows you should use Everything program that mimic locate, is very fast and don't need to update database manually. To use with helm-locate, you will need his command line named "es". Be sure to modify the PATH environment variable, to include path to the directory that contain "es". The arguments are the same than the ones in "locate".

11 Helm Etags

11.1 Create the tag file

To use etags in Emacs you have first to create a TAGS file for your project with the etags shell command. If your directory contains subdirectories use someting like :(e.g .el files)

```
find . -iregex .*\.el$ | xargs etags
```

Otherwise

```
etags *.el
```

is enough

For more infos see the man page of etags.

11.2 Start helm etags

Now just using f5 a e will show you all entries. If the project is big it take some time to load tag file but when it is done, next search will be very fast. If you modify the TAGS file, use

```
C-u C-u f5 a e
```

to refresh the tag cache.

To search the definition at point use just

```
C-u f5 a e
```

12 Firefox bookmarks

You will have to set firefox to import bookmarks in his html file bookmarks.html.

`(only for firefox versions >=3)`

To achieve that, open "about :config" in firefox and double click on this line to enable value to true :

```
user_pref("browser.bookmarks.autoExportHTML", false);
```

You should have now :

```
user_pref("browser.bookmarks.autoExportHTML", true);
```

Now you can use

`M-x helm-firefox-bookmarks`

To see your firefox bookmarks from Emacs. When you are in firefox things are a little more complicated. You will need wmctrl program and a script named ffbookmarks :

```
#!/bin/bash

wmctrl -xa emacs
emacsclient -e "(progn (helm-firefox-bookmarks) nil)" > /dev/null
wmctrl -xa firefox
exit 0
```

Put this script somewhere in PATH and make it executable :

`chmod +x ffbookmarks`

Firefox is not aware about this new protocol, you will have to instruct it. See Firefox documentation or use firefox-protocol.el package you can get here :
http://mercurial.intuxication.org/hg/emacs-bookmark-extension/
Install new protocol :

`M-x firefox-protocol-installer-install`

to install new protocol ffbookmarks Then install a bookmarklet in firefox : Right click on the bookmark toolbar in firefox and add a new bookmark called ffbookmarks. Add this instead of url :

```
javascript:location.href='ffbookmarks://localhost'
```

Now when you click on ffbookmarks it will bring you in Emacs and allow you to browse your bookmarks with helm.
NOTE : emacs server need to be started in the running Emacs, see Emacs documentation.

13 Helm for buffers

`M-x helm-buffers-list`

13.1 Borring buffers

Will show you your buffers list without borring buffers defined by regexp in "helm-c-boring-buffer-regexp". Just use customize to set that for your need.

13.2 Search buffers by major-mode

Once in this helm session, you can narrow your buffer list by "major mode", regexp as usual or the both :

Example :

I want to show all my buffer that are in emacs-lisp mode :

Pattern : lisp

will show all emacs-lisp and lisp related buffers.

Now i want to limit these buffers to the one that match "any"

Pattern : lisp any

I want to match buffers that match "any" but not limited to lisp buffers :

Pattern : any

13.3 Different colors for buffers

- If a buffer is modified, it will showup in orange.
- If a buffer have been modified by some external program (e.g sed) in the back of emacs, it will showup in red.
- Non–buffer file, Directory and files have differents face.

13.4 Special commands

Not complete.

C-c ? will show you all commands available.

14 Other tools

In addition of what is described above, you will find a bunch of powerfull tools that come with helm-config.el. Just browse the helm commands availables with helm-M-x.

Not complete.

15 Helm completion mode

'helm-completion-mode' aka 'ac-mode' will enable helm completion in all Emacs commands using 'completing-read' or 'read-file-name'.
To use it :

```
M-x ac-mode
```

Turn it on in .emacs with :

```
(ac-mode 1)
```

Customize with :
'helm-completing-read-handlers-alist'

See C-h v 'helm-completing-read-handlers-alist' for more infos.

Not complete.

16 Helm Eshell completion

16.1 Enable helm pcomplete

Of course pcomplete is already enabled in Eshell, but what we want here is to enable it with helm support.
To enable it's easy, just add to ".emacs.el" :

```
(add-hook 'eshell-mode-hook
          #'(lambda ()
              (define-key eshell-mode-map
                [remap pcomplete]
                'helm-esh-pcomplete)))
```

Now when hitting "TAB", you should have helm pcompletion.

16.1.1 Write your own pcomplete functions

You can enhance Emacs pcomplete by writing your own pcomplete functions.

Here an example with "find" command :

```
(defun pcomplete/find ()
  (let ((prec (pcomplete-arg 'last -1)))
    (cond ((and (pcomplete-match "^-" 'last)
                (string= "find" prec))
           (pcomplete-opt "HLPDO"))
          ((pcomplete-match "^-" 'last)
           (while (pcomplete-here
                   '("-amin" "-anewer" "-atime" "-cmin" "-cnewer" "-context"
                     "-ctime" "-daystart" "-delete" "-depth" "-empty" "-exec"
                     "-execdir" "-executable" "-false" "-fls" "-follow" "-fprint"
                     "-fprint0" "-fprintf" "-fstype" "-gid" "-group"
                     "-help" "-ignore_readdir_race" "-ilname" "-iname"
                     "-inum" "-ipath" "-iregex" "-iwholename"
                     "-links" "-lname" "-ls" "-maxdepth"
                     "-mindepth" "-mmin" "-mount" "-mtime"
                     "-name" "-newer" "-nogroup" "-noignore_readdir_race"
                     "-noleaf" "-nouser" "-nowarn" "-ok"
                     "-okdir" "-path" "-perm" "-print"
                     "-print0" "-printf" "-prune" "-quit"
                     "-readable" "-regex" "-regextype" "-samefile"
                     "-size" "-true" "-type" "-uid"
                     "-used" "-user" "-version" "-warn"
                     "-wholename" "-writable" "-xdev" "-xtype"))))
          ((string= "-type" prec)
           (while (pcomplete-here (list "b" "c" "d" "p" "f" "l" "s" "D"))))
          ((string= "-xtype" prec)
           (while (pcomplete-here (list "b" "c" "d" "p" "f" "l" "s"))))
          ((or (string= prec "-exec")
               (string= prec "-execdir"))
           (while (pcomplete-here* (funcall pcomplete-command-completion-function)
                                   (pcomplete-arg 'last) t))))
    (while (pcomplete-here (pcomplete-entries) nil 'identity))))
```

16.2 Enable helm Eshell history

Add this to ".emacs.el"

```
(add-hook 'eshell-mode-hook
          #'(lambda ()
              (define-key eshell-mode-map
                (kbd "M-p")
                'helm-eshell-history)))
```

17 Usefuls extensions

Not complete.

18 Usefuls links

You can have infos about helm on Github.
https://github.com/emacs-helm/helm

You can ask on the helm mailing-list by subscribing at :
https://groups.google.com/group/emacs-helm?hl=en
Or at gmane : gmane.emacs.helm