NAME

clisp – **ANSI**[38] **Common Lisp**[1] compiler, interpreter and debugger.

SYNOPSIS

clisp [[**−h**] | [**−−help**]] [**−−version**] [**−−license**] [**−help−image**] [**−B** *lisp−lib−dir*] [**−b**] [**−K** *linking−set*] [**−M** *mem−file*] [**−m** *memory−size*] [**−L** *language*] [**−N** *locale−dir*] [**−E***domain encoding*] [[**−q**] | [**−−quiet**] | [**−−silent**] | [**−v**] | [**−−verbose**]] [**−on−error** *action*] [**−repl**] [**−w**] [**−I**] [**−disable−readline**] [[**−ansi**] | [**−traditional**]] [**−modern**] [**−p** *package*] [**−C**] [**−norc**] [**−lp** *directory*...] [**−i** *init−file*...] [**−c** [**−l**] *lisp−file* [**−o** *output−file*]...] [**−x** *expressions*...] [*lisp−file* [*argument*...]]

DESCRIPTION

Invokes the **Common Lisp**[1] interpreter and compiler.

Interactive Mode

When called without batch arguments, executes the **read−eval−print loop**[2], in which expressions are in turn

- **READ**[3] from the standard input,
- **EVAL**[4]uated by the lisp interpreter,
- and their results are **PRINT**[5]ed to the standard output.

Non−Interactive (Batch) Mode

Invoked with **−c**, compiles the specified lisp files to a platform−independent bytecode which can be executed more efficiently.

Invoked with **−x**, executes the specified lisp expressions.

Invoked with *lisp−file*, runs the specified lisp file.

OPTIONS

−h

−−help

Displays a help message on how to invoke **CLISP**[6].

−−version

Displays the **CLISP**[6] version number, as given by the function **LISP−IMPLEMENTATION−VERSION**[7], the value of the variable **FEATURES**, as well some other information.

−−license

Displays a summary of the licensing information, the **GNU**[8] **GPL**[9].

−help−image

Displays information about the memory image being invoked: whether is it suitable for scripting as well as the **:DOCUMENTATION** supplied to **EXT:SAVEINITMEM**.

−B *lisp−lib−dir*

Specifies the installation directory. This is the directory containing the linking sets and other data files. This option is normally not necessary, because the installation directory is already built−in into the **clisp** executable. Directory *lisp−lib−dir* can be changed dynamically using the **SYMBOL−MACRO**[10] *CUSTOM:*LIB−DIRECTORY**.

−b

Print the installation directory and exit immediately. The namestring of *CUSTOM:*LIB−DIRECTORY** is printed without any quotes. This is mostly useful in module Makefiles, see, e.g., modules/syscalls/Makefile.in (file in the CLISP sources).

−K *linking−set*

Specifies the linking set to be run. This is a directory (relative to the *lisp−lib−dir*) containing at least a main executable (runtime) and an initial memory image. Possible values are

base

the core **CLISP**[6]

full
core plus all the modules with which this installation was built, see Section 32.2, “External Modules”.

The default is **base**.

–M *mem–file*
Specifies the initial memory image. This must be a memory dump produced by the **EXT:SAVEINITMEM** function by this **clisp** runtime. It may have been compressed using **GNU**[8] **gzip**[11].

–m *memory–size*
Sets the amount of memory **CLISP**[6] tries to grab on startup. The amount may be given as

n
*n***B**
measured in bytes

n
*n***W**
measured in machine words (4×*n* on 32–bit platforms, 8×*n* on 64–bit platforms)

*n***K**
*n***KB**
measured in kilobytes

*n***KW**
measured in kilowords

*n***M**
*n***MB**
measured in megabytes

*n***MW**
measured in megawords

The default is 3 megabytes. The argument is constrained above 100 KB.

This version of **CLISP**[6] is not likely to actually use the entire *memory–size* since garbage–collection will periodically reduce the amount of used memory. It is therefore common to specify 10 MB even if only 2 MB are going to be used.

–L *language*
Specifies the language **CLISP**[6] uses to communicate with the user. This may be one of **english**, **german**, **french**, **spanish**, **dutch**, **russian**, **danish**. Other languages may be specified through the **environment variable**[12] **LANG**, provided the corresponding message catalog is installed. The language may be changed dynamically using the **SYMBOL–MACRO**[10] *CUSTOM:*CURRENT–LANGUAGE**.

–N *locale–dir*
Specifies the base directory of locale files. **CLISP**[6] will search its message catalogs in *locale–dir*/*language*/LC_MESSAGES/clisp.mo. This directory may be changed dynamically using the **SYMBOL–MACRO**[10] *CUSTOM:*CURRENT–LANGUAGE**.

–E*domain encoding*
Specifies the encoding used for the given domain, overriding the default which depends on the **environment variable**[12]s **LC_ALL**, **LC_CTYPE**, **LANG**. *domain* can be

file
affecting *CUSTOM:*DEFAULT–FILE–ENCODING**

pathname
affecting *CUSTOM:*PATHNAME−ENCODING**

terminal
affecting *CUSTOM:*TERMINAL−ENCODING**

foreign
affecting *CUSTOM:*FOREIGN−ENCODING**

misc
affecting *CUSTOM:*MISC−ENCODING**

blank
affecting all of the above.

Warning

Note that the values of these **SYMBOL−MACRO**[10]s that have been saved in a memory image are ignored: these **SYMBOL−MACRO**[10]s are reset based on the OS environment **after** the memory image is loaded. You have to use the RC file, *CUSTOM:*INIT−HOOKS** or init function to set them on startup, but it is best to set the aforementioned **environment variable**[12]s appropriately for consistency with other programs. See Section 31.1, “Customizing CLISP Process Initialization and Termination”.

−q
−−quiet
−−silent
−v
−−verbose
Change verbosity level: by default, **CLISP**[6] displays a banner at startup and a good−bye message when quitting, and initializes **LOAD−VERBOSE**[13] and **COMPILE−VERBOSE**[14] to **T**[15], and **LOAD−PRINT**[13] and **COMPILE−PRINT**[14] to **NIL**[16], as per [ANSI CL standard]. The first **−q** removes the banner and the good−bye message, the second sets variables **LOAD−VERBOSE**[13], **COMPILE−VERBOSE**[14] and *CUSTOM:*SAVEINITMEM−VERBOSE** to **NIL**[16]. The first **−v** sets variables *CUSTOM:*REPORT−ERROR−PRINT−BACKTRACE**, **LOAD−PRINT**[13] and **COMPILE−PRINT**[14] to **T**[15], the second sets *CUSTOM:*LOAD−ECHO** to **T**[15]. These settings affect the output produced by **−i** and **−c** options. Note that these settings persist into the **read−eval−print loop**[2]. Repeated **−q** and **−v** cancel each other, e.g., **−q −q −v −v −v** is equivalent to **−v**.

−on−error *action*
Establish global error handlers, depending on *action*:.PP appease
continuable[17] **ERROR**[18]s are turned into **WARNING**[19]s (with **EXT:APPEASE−CERRORS**) other **ERROR**[18]s are handled in the default way

debug
ERROR[18]s **INVOKE−DEBUGGER**[20] (the normal **read−eval−print loop**[2] behavior), disables batch mode imposed by **−c**, **−x**, and *lisp−file*,

abort
continuable[17] **ERROR**[18]s are appeased, other **ERROR**[18]s are **ABORT**[21]ed with **EXT:ABORT−ON−ERROR**

exit
continuable[17] **ERROR**[18]s are appeased, other **ERROR**[18]s terminate **CLISP**[6] with **EXT:EXIT−ON−ERROR** (the normal batch mode behavior).

See also **EXT:SET−GLOBAL−HANDLER**.

−repl
Start an interactive **read−eval−print loop**[2] after processing the **−c**, **−x**, and *lisp−file* options and on any **ERROR**[18] **SIGNAL**[22]ed during that processing.

Disables batch mode.

−w
Wait for a keypress after program termination.

−I
Interact better with **Emacs**[23] (useful when running **CLISP**[6] under **Emacs**[23] using **SLIME**[24], **ILISP**[25] et al). With this option, **CLISP**[6] interacts in a way that **Emacs**[23] can deal with:

- unnecessary prompts are not suppressed.
- The **GNU**[8] **readline**[26] library treats TAB (see TAB key) as a normal self−inserting character (see Q: A.4.6).

−disable−readline
Do not use **GNU**[8] **readline**[26] even when it has been linked against. This can be used if one wants to paste non−*ASCII*[27] characters, or when **GNU**[8] **readline**[26] misbehaves due to installation (different versions on the build and install machines) or setup (bad **TERM environment variable**[12] value) issues.

−ansi
Comply with the [ANSI CL standard] specification even where **CLISP**[6] has been traditionally different by setting the **SYMBOL−MACRO**[10] *CUSTOM:*ANSI** to **T**[15].

−traditional
Traditional: reverses the residual effects of **−ansi** in the saved memory image.

−modern
Provides a modern view of symbols: at startup the **PACKAGE**[28] variable will be set to the "CS−COMMON−LISP−USER" package, and the **PRINT−CASE**[29] will be set to **:DOWNCASE**. This has the effect that symbol lookup is case−sensitive (except for keywords and old−style packages) and that keywords and uninterned symbols are printed with lower−case preferrence. See Section 11.5, "Package Case-Sensitivity".

−p *package*
At startup the value of the variable **PACKAGE**[28] will be set to the package named *package*. The default is the value of **PACKAGE**[28] when the image was saved, normally **"COMMON−LISP−USER"**[30].

−C
Compile when loading: at startup the value of the variable *CUSTOM:*LOAD−COMPILING** will be set to **T**[15]. Code being **LOAD**[31]ed will then be **COMPILE**[32]d on the fly. This results in slower loading, but faster execution.

−norc
Normally **CLISP**[6] loads the user **"run control" (RC)**[33] file on startup (this happens **after** the **−C** option is processed). The file loaded is .clisprc.lisp or .clisprc.fas in the home directory **USER−HOMEDIR−PATHNAME**[34], whichever is newer. This option, **−norc**, prevents loading of the RC file.

−lp *directory*
Specifies directories to be added to *CUSTOM:*LOAD−PATHS** at startup. This is done **after** loading the RC file (so that it does not override the command−line option) but **before** loading the init−files specified by the **−i** options (so that the init−files will be searched for in the specified directories). Several **−lp** options can be given; all the specified directories will be added.

−i *init−file*
Specifies initialization files to be **LOAD**[31]ed at startup. These should be lisp files (source or compiled). Several **−i** options can be given; all the specified files will be loaded in order.

−c *lisp−file*
Compiles the specified *lisp−file*s to bytecode (*.fas). The compiled files can then be **LOAD**[31]ed instead of the sources to gain efficiency.

Imposes batch mode.

−o *outputfile*
Specifies the output file or directory for the compilation of the last specified *lisp−file*.

−l
Produce a bytecode **DISASSEMBLE**[35] listing (*.lis) of the files being compiled. Useful only for debugging. See Section 24.1, “Function COMPILE-FILE” for details.

−x *expressions*
Executes a series of arbitrary expressions instead of a **read−eval−print loop**[2]. The values of the expressions will be output to **STANDARD−OUTPUT**[36]. Due to the argument processing done by the shell, the *expressions* must be enclosed in double quotes, and double quotes and backslashes must be escaped with backslashes.

Imposes batch mode.

lisp−file [*argument* ...]
Loads and executes a *lisp−file*, as described in Section 32.6.2, “Scripting with CLISP”. There will be no **read−eval−print loop**[2]. Before *lisp−file* is loaded, the variable *EXT:*ARGS** will be bound to a list of strings, representing the *argument*s. The first line of *lisp−file* may start with **#!**, thus permitting **CLISP**[6] to be used as a script interpreter. If *lisp−file* is **−**, the **STANDARD−INPUT**[36] is used instead of a file.

This option is *disabled* if the memory image was created by **EXT:SAVEINITMEM** with **NIL**[16] **:SCRIPT** argument. In that case the **LIST**[37] *EXT:*ARGS** starts with *lisp−file*.

This option must be the last one.

No RC file will be executed.

Imposes batch mode.

As usual, **−−** stops option processing and places all remaining command line arguments into *EXT:*ARGS**.

LANGUAGE REFERENCE

The language implemented is **ANSI**[39][38] **Common Lisp**[1]. The implementation mostly conforms to the ANSI Common Lisp standard, see Section 31.10, “Maximum ANSI CL compliance”. [ANSI CL] ANSI CL standard1994. ANSI[40] INCITS 226-1994 (R1999)
Information Technology - Programming Language - Common Lisp
[formerly ANSI X3.226-1994 (R1999)].

COMMAND LINE USER ENVIRONMENT

help
get context−sensitive on−line help, see Chapter 25, Environment chap-25.

(**APROPOS** *name*)
list the **SYMBOL**[41]s matching *name*.

(**DESCRIBE** *symbol*)
describe the *symbol*.

(exit)
(quit)
(bye)
quit **CLISP**[6].

EOF (Control+D on **UNIX**[42])
leave the current level of the **read−eval−print loop**[2] (see also Section 1.1, “Special Symbols sec_1-4-1-3”).

arrow keys

for editing and viewing the input history, using the **GNU**[8] **readline**[26] library.

TAB key
Context sensitive:

- If you are in the "function position" (in the first symbol after an opening paren or in the first symbol after a **#'**[44]), the completion is limited to the symbols that name functions.
- If you are in the "filename position" (inside a string after **#P**[45]), the completion is done across file names, **GNU**[8] **bash**[46]–style.
- If you have not typed anything yet, you will get a help message, as if by the **help** command.
- If you have not started typing the next symbol (i.e., you are at a whitespace), the current function or macro is **DESCRIBE**d.
- Otherwise, the symbol you are currently typing is completed.

USING AND EXTENDING CLISP

Common Lisp[1] is a *programmable* programming language. —**John Foderaro**[47].PP When **CLISP**[6] is invoked, the runtime loads the initial memory image and outputs the prompt; at which one can start typing **DEFVAR**[48]s, **DEFUN**[49]s and **DEFMACRO**[50]s.

To avoid having to re–enter the same definitions by hand in every session, one can create a lisp file with all the variables, functions, macros, etc.; (optionally) compile it with **COMPILE–FILE**[51]; and **LOAD**[31] it either by hand or from the RC file; or save a memory image to avoid the **LOAD**[31] overhead.

However, sometimes one needs to use some functionality implemented in another language, e.g., call a **C**[52] library function. For that one uses the Foreign Function Interface and/or the External Modules facility. Finally, the truly adventurous ones might delve into Extending the Core.

FILES

clisp
clisp.exe
startup driver (an executable or, rarely, a shell script) which remembers the location of the runtime and starts it with the appropriate arguments

lisp.run
lisp.exe
main executable (runtime) – the part of **CLISP**[6] implemented in **C**[52].

lispinit.mem
initial memory image (the part of **CLISP**[6] implemented in lisp)

config.lisp
site–dependent configuration (should have been customized before **CLISP**[6] was built); see Section 31.12, "Customizing CLISP behavior"

*.lisp
lisp source

*.fas
lisp code, compiled by **CLISP**[6]

*.lib
lisp source library information, generated by **COMPILE–FILE**, see Section 24.3, "Function REQUIRE".

*.c
C code, compiled from lisp source by **CLISP**[6] (see Section 32.3, "The Foreign Function Call Facility")

For the **CLISP**[6] source files, see Chapter 34, The source files of CLISP.

ENVIRONMENT

All **environment variable**[12]s that **CLISP**[6] uses are read at most once.

CLISP_LANGUAGE
specifies the language **CLISP**[6] uses to communicate with the user. The legal values are identical to those of the **−L** option which can be used to override this **environment variable**[12].

LC_CTYPE
specifies the locale which determines the character set in use. The value can be of the form *language* or *language_country* or *language_country.charset*, where *language* is a two−letter ISO 639 language code (lower case), *country* is a two−letter ISO 3166 country code (upper case). *charset* is an optional character set specification, and needs normally not be given because the character set can be inferred from the language and country. This **environment variable**[12] can be overridden with the **-Edomain encoding** option.

LANG
specifies the language **CLISP**[6] uses to communicate with the user, unless it is already specified through the **environment variable**[12] **CLISP_LANGUAGE** or the **−L** option. It also specifies the locale determining the character set in use, unless already specified through the **environment variable**[12] **LC_CTYPE**. The value may begin with a two−letter ISO 639 language code, for example **en**, **de**, **fr**.

HOME
USER
used for determining the value of the function **USER−HOMEDIR−PATHNAME**[34].

SHELL
COMSPEC
is used to find the interactive command interpreter called by **EXT:SHELL**.

TERM
determines the screen size recognized by the pretty printer.

ORGANIZATION
for **SHORT−SITE−NAME**[53] and **LONG−SITE−NAME**[53] in config.lisp.

CLHSROOT
for **CUSTOM:CLHS−ROOT** in config.lisp.

IMPNOTES
for **CUSTOM:IMPNOTES−ROOT** in config.lisp.

EDITOR
for **editor−name** in config.lisp.

LOGICAL_HOST_*host*_**FROM**
LOGICAL_HOST_*host*_**TO**
LOGICAL_HOST_*host*
for *CUSTOM:*LOAD−LOGICAL−PATHNAME−TRANSLATIONS−DATABASE**

INPUT AND OUTUT

See Section 21.1.1, “Initialization of Standard Streams”.

SEE ALSO

CLISP impnotes
clisp-link(1)
CMU CL[54] **– cmucl**(1)
SBCL[55] **– sbcl**(1)
Emacs[23] **– emacs**(1)

BUGS

When you encounter a bug in **CLISP**[6] or in its documentation (this manual page or CLISP impnotes), please report it to the **CLISP**[6] **SourceForge bug tracker**[56]. Visit this bug tracker with a browser other

than Firefox (because as of May 2017, the "Create Ticket" button is not visible in Firefox). Then login, either to your **SourceForge**[57] account, or to your **OpenID**[58] account. Then press the "Create Ticket" button on the left−hand side.

Before submitting a bug report, please take the following basic steps to make the report more useful:

1. Unless your bug is locale−specific, please set your locale to en. You *cannot* assume that **CLISP**[6] maintainers understand a language other than **English**[59], even though, historically, few **CLISP**[6] maintainers spoke English natively.
2. Do a clean build (remove your build directory and build **CLISP**[6] with **./configure −−cbc build** or at least do a **make distclean** before **make**).
3. If you are reporting a "hard crash" (segmentation fault, bus error, core dump etc), please do **./configure −−with−debug −−cbc build−g ; cd build−g; gdb lisp.run**, then load the appropriate linking set by either **base** or **full gdb**[60] command, and report the backtrace (see also Q: A.1.1.10).
4. If you are using pre−built binaries and experience a hard crash, the problem is likely to be in the incompatibilities between the platform on which the binary was built and yours; please try compiling the sources and report the problem if it persists.

When submitting a bug report, please specify the following information:

1. What is your platform (**uname −a** on a **UNIX**[42] system)? Compiler version? **GNU**[8] **libc**[61] version (on **GNU**[8]/*Linux*[62])?
2. Where did you get the sources or binaries? When? (Absolute dates, e.g., "2006−01−17", are preferred over the relative ones, e.g., "2 days ago". If you are using **Git**[63], please supply the output of **git rev-list --max-count=1 HEAD**).
3. How did you build **CLISP**[6]? (What command, options &c.)
4. What is the output of **clisp −−version**?
5. Please supply the full output (copy and paste) of all the error messages, as well as detailed instructions on how to reproduce them.

PROJECTS

- Enhance the compiler so that it can inline local functions.
- Embed **CLISP**[6] in **VIM**[64].

AUTHORS

Bruno Haible <http://www.haible.de/bruno/>
The original author and long−time maintainer.

Michael Stoll <http://www.mathe2.uni-bayreuth.de/stoll/>
The original author.

Sam Steingold <http://sds.podval.org/>
Co−maintainer since 1998.

Others
See *COPYRIGHT (file in the CLISP sources)* for the list of other contributors and the license.

COPYRIGHT

NOTES

1. **Common Lisp**
   https://common-lisp.net
2. read-eval-print loop
   [set $man.base.url.for.relative.links]/sec_25-1-1

3. **READ**
   http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_readcm_re_g-whitespace.html
4. **EVAL**
   http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_eval.html
5. **PRINT**
   http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_writecm_p_rintcm_princ.html
6. **CLISP**
   http://clisp.org
7. **LISP-IMPLEMENTATION-VERSION**
   http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_lisp-impl_tion-version.html
8. GNU
   https://www.gnu.org
9. GPL
   https://www.gnu.org/copyleft/gpl.html
10. SYMBOL-MACRO
    [set $man.base.url.for.relative.links]/mac_define-symbol-macro
11. **gzip**
    http://www.gzip.org/
12. environment variable
    [set $man.base.url.for.relative.links]/basedefs/xbd_chap08.html
13. **LOAD-VERBOSE**
    http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/var_stload-pr_ad-verbosest.html
14. **COMPILE-VERBOSE**
    http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/var_stcompile_le-verbosest.html
15. **T**
    http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/convar_t.html
16. **NIL**
    http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/convar_nil.html
17. continuable
    [set $man.base.url.for.relative.links]/clhs/glo
18. ERROR
    http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/contyp_error.html
19. WARNING
    http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/contyp_warning.html
20. **INVOKE-DEBUGGER**
    http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_invoke-debugger.html
21. **ABORT**
    http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_abortcm_c_cm_use-value.html
22. **SIGNAL**
    http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_signal.html
23. Emacs
    https://www.gnu.org/software/emacs/
24. SLIME
    https://common-lisp.net/project/slime/
25. ILISP
    https://sourceforge.net/projects/ilisp/

26. readline
http://tiswww.case.edu/php/chet/readline/readline.html
27. *ASCII*
https://en.wikipedia.org/wiki/ASCII
28. **PACKAGE**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/var_stpackagest.html
29. **PRINT-CASE**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/var_stprint-casest.html
30. "COMMON-LISP-USER"
[set $man.base.url.for.relative.links]/sec_11-1-2-2
31. **LOAD**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_load.html
32. **COMPILE**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_compile.html
33. "run
control" (RC)
http://www.faqs.org/docs/artu/ch10s03.html
34. **USER-HOMEDIR-PATHNAME**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_user-homedir-pathname.html
35. **DISASSEMBLE**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_disassemble.html
36. **STANDARD-OUTPUT**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/var_stdebug-i_ace-outputst.html
37. LIST
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/syscla_list.html
38. ANSI
https://www.ansi.org/
39. The American National Standards Institute
40. ANSI
https://webstore.ansi.org
41. SYMBOL
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/syscla_symbol.html
42. **UNIX**
http://www.unix.org/online.html
43. *Win32*
https://winehq.org/
44. #'
[set $man.base.url.for.relative.links]/sec_2-4-8-2
45. #P
[set $man.base.url.for.relative.links]/sec_2-4-8-14
46. bash
https://www.gnu.org/software/bash/
47. John Foderaro
http://www.franz.com/~jkf/
48. **DEFVAR**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/mac_defparametercm_defvar.html

49. **DEFUN**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/mac_defun.html
50. **DEFMACRO**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/mac_defmacro.html
51. **COMPILE-FILE**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_compile-file.html
52. **C**
http://c-faq.com/
53. **SHORT-SITE-NAME**
http://www.ai.mit.edu/projects/iiip/doc/CommonLISP/HyperSpec/Body/fun_short-sit_ng-site-name.html
54. **CMU CL**
https://www.cons.org/cmucl/
55. **SBCL**
http://www.sbcl.org/
56. SourceForge bug tracker
https://sourceforge.net/p/clisp/bugs/
57. SourceForge
https://sourceforge.net
58. OpenID
https://openid.net/
59. English
http://www.catb.org/esr/faqs/hacker-howto.html#skills4
60. **gdb**
https://www.sourceware.org/gdb/
61. libc
https://www.gnu.org/software/libc/
62. *Linux*
https://www.kernel.org/
63. Mercurial
https://www.mercurial-scm.org/
64. VIM
https://www.vim.org