CS3101 Programming Languages (Lisp)

Installing Lisp & Emacs Cheat Sheet

Bob Coyne (coyne@columbia.edu)

Columbia University

March 10, 2017

Recommended configurations

See instructions (below) for each of the following configurations. You should install at least one of these, though it can be useful to try more than one to compare.

- Running Lispworks (locally). Lispworks has a built-in Emacs-like editor.
- Running SBCL/Emacs/Slime (locally)
- Running SBCL/Emacs/Slime (remotely) on Cunix

Note: In all configs, we also install Quicklisp, to provide access to external Lisp libraries.

Installing Lispworks

Lispworks is a commercial Lisp with a free downloadable version. It includes an Emacs-like editor and Lisp-based development tools.

Download and install Lispworks
`http://www.lispworks.com/downloads/`

Download and install Quicklisp: `https://www.quicklisp.org/beta/`
$ curl -O https://beta.quicklisp.org/quicklisp.lisp
Start Lispworks, then evaluate the following Lisp forms

∗ (load "~/quicklisp.lisp")
∗ (quicklisp-quickstart:install)
∗ (ql:add-to-init-file)

Installing Emacs, SBCL, Quicklisp, Slime (locally)

```
Install Emacs (v24 or newer). Note: some Macs have outdated Emacs.
  Mac: https://emacsformacosx.com/
  Windows: http://mirrors.ocf.berkeley.edu/gnu/emacs/windows/

Install SBCL. Find versions here: (not all are binary)
  https://sourceforge.net/projects/sbcl/files/sbcl/
  ** For Macs: Use newest MacOS binary: sbcl-1.2.11-x86-64-darwin

Install Quicklisp and Slime (follow instructions) here:
  https://www.quicklisp.org/beta/
```

Note: after going through this process, you will have the following config files/dirs:

˜/.emacs [** for Emacs]
˜/.sbclrc [** for SBCL]
˜/.slime/ [** for Slime]
˜/quicklisp/ [** for Quicklisp]

Installing Quicklisp and Slime (Cunix)

In this config you run Emacs and SBCL (already installed) remotely on Cunix. You'll need X-Windows installed locally.
On MacOS, install: xquartz (x-windows) https://www.xquartz.org/

Configure Emacs and SBCL and install Quicklisp and Slime as follows:

Connect to cunix: ssh -X username@cunix.columbia.edu
For Windows, use Putty to connect to Cunix: http://www1.cs.columbia.edu/~bert/courses/1003/cunix.html#Getting%20to%20Cunix

```
$ curl -O https://beta.quicklisp.org/quicklisp.lisp
$ sbcl –load quicklisp.lisp
∗ (quicklisp-quickstart:install)
∗ (ql:add-to-init-file)
∗ (quit)
$ cp -R /u/1/r/rc2447/.slime ˜/.slime [Slime version compatible with Emacs v23]
$ cp /u/1/r/rc2447/.emacs.cs3101 ˜/.emacs [Configures Emacs to use Slime]
```

Emacs meta key on Mac (for Cunix without X-forwarding)

Emacs heavily uses the Meta key. Mac keyboards have no meta key. Emacs will often remap the Option key to Meta.

However if running Emacs remotely in the Terminal (i.e. not via X-Windows), that remapping won't happen by default. The following lets you use Option Key as Meta.

`https://www.emacswiki.org/emacs/EmacsForMacOS`

- If possible, use X-Forwarding when running remotely: ssh -X...
- Else, run Emacs in Terminal and tell Mac to use Alt/Option as Meta Key

  For Terminal.app:
  - starting from Snow Leopard: Go to Preferences > Settings > Keyboard > Use option as meta key.
  - Starting from El Capitan (or maybe earlier): Go to Preferences > Profiles > Keyboard > Use Option as Meta Key.

  For iTerm: Go to Manage Profiles > Keyboard profiles > (your profile) > Option Key as...

  For iTerm2: Go to Preferences > Profiles > Keys > Left/Right option key acts as: Meta

Running Lisp

SBCL on Cunix (remotely), first connect:
ssh -X YOUR_UNI@cunix.columbia.edu

- start emacs at shell with: emacs
- Start SBCL lisp within Slime: m-x slime
- You can now use Lisp in Emacs

SBCL locally

- Launch Emacs. Check version is at least 24 (m-x version)
- Start SBCL lisp within Slime: m-x slime
- You can now use Lisp in Emacs

Lispworks (locally)

- Launch Lisworks. Use Editor and Listener Windows.
- Note: compiling a Lisp form in Lispworks editor is c-sh-c (not c-c c-c as in regular Emacs).

Getting started

To write lisp code you generally have a buffer or buffers with your code and use emacs commands to compile it. You switch to the REPL to run/test as you work.

To create a buffer, type c-x c-f to emacs (or the Lispworks editor). Give your file a name that ends in ".lisp" (eg "my-code.lisp"). This will tell the editor that you are editing lisp vs random text. You can save your file with c-x c-s or write it to a new location with c-x c-w.

Put a package declaration at the top of the file. For now, just use (in-package :cl-user).

Then add your code below. Remember to keep it formatted/indented (using c-m-q). You can use c-c c-c (in GNU Emacs) or c-sh-c (in the Lispworks editor) to compile the Lisp form under the cursor, which will often point out errors.

See Emacs Cheat Sheet (next slide) and online documentation for more.

GNU Emacs Cheat Sheet (Lispworks editor mostly compatible)

Tour: https://www.gnu.org/software/emacs/tour/

Emacs Help
- Tutorial: c-h t
- Index of help commands: c-h ?
- Search for command: c-h a
- Describe command: c-h w
- Describe command: m-x apropos
- Describe key binding: c-h c

General
- Abort command (eg search): c-g
- Search: c-s (forward), c-r (rev)
- Search&repl: m-% [type values]
- Mark region: c-SPACE and move
- Exit: c-x c-c
- Load E-lisp file: m-x load-file
- Eval E-lisp expr: m-:

Panes
- Two panes (horiz): c-x
- Two panes (vert): c-x 2
- Single pane (selected one): c-x 0
- Switch focus to other pane: c-x o

Files and buffers
- Select buffer: c-x b [type name]
- List buffers: c-x c-b
- Open file: c-x c-f [type name]
- Save: c-x c-s
- Save as: c-x c-w [type name]
- Kill buffer: c-x k [select buffer]
- Create shell buffer: m-x shell

Lisp
- Start Lisp: m-x slime
- Compile Lisp form: c-c c-c
- Compile Lisp buffer: c-c c-k
- Eval Lisp form: c-m-x
- Format/indent s-expr: c-m-q
- Symbol complete c-m-i, c-c TAB
- Find Lisp definition: m-.
- TAB will indent or complete
- SPACE will show function arglists
- REPL buffer: *slime-repl sbcl*
- Debugger buffer: *sldb sbcl/0*
- Yank prev REPL cmd: m-p
- Yank next REPL cmd: c-p

Deleting/restoring
- Char: c-d
- Word: m-d
- S-expr: c-m-d
- Kill line (store): c-k
- Delete/store marked region: c-w
- Store marked region: m-w
- Yank (paste) stored text: c-y
- Undo: c-/ or c-_

Misc text
- Upper case: m-u
- Lowercase: m-l
- Open newline: c-o
- Transpose s-expressions: c-m-t

Move cursor

```
c-f Forward a character
c-b Backward a character
m-f Forward a word
m-b Backward a word
c-m-f Forward s-expression
c-m-b Backward s-expression
c-c c-p Move to prev REPL cmd
c-c c-n Move to next REPL cmd

c-n Next line
c-p Previous line
c-a Beginning of line
c-e End of line

c-a Beginning of s-expression
c-e End of s-expression
c-m-u   UP s-expression

m-v Backward a page
c-v forward a page
c-l Center on page
m-< File beginning
m-> File end
```

Useful extensions
- Swap current buffer: c-;
- Select REPL buffer: c-m-;
- Select shell buffer: c-,
- Comment region: c-=
- Uncomment region: c-&