Getting up and running with Emacs and Common Lisp
INF4820, Fall 2012

For our practical exercises in INF4820, we will be using Allegro Common Lisp (ACL), which is installed on all Linux machines at IFI. Here we describe how to configure your set-up and connect to a server to run Common Lisp in the Emacs editor.

1 How to connect

Most of the **IFI student laboratories** are installed with Linux, and here it's simply a matter of logging in on the desktop (and you're ready to skip to Section 2). But if you're working from your own machine or from a Windows installation you'll need to connect to a Linux machine and use the X Window System to allow remote applications to display on the local screen.

If you're using an **IFI machine running Windows**, the standard installation should include a desktop shortcut (labeled 'xterm') to launch an X server and connect to a Linux machine. If your instead working from home **using your own machine** you can try one of the following options:

– Option 1: Using a Windows remote desktop session, connect to '`windows.ifi.uio.no`' and then simply follow the instructions given for IFI Windows machines above. Note that Windows remote desktop clients are available for Linux and Mac too. You'll find more information for all platforms here:
English: `http://www.uio.no/english/services/it/network/home-and-away/rdp/`
Norwegian: `http://www.uio.no/tjenester/it/nett/utenfra/rdp/`

– Option 2: From a command line / terminal, use ssh to connect to a IFI Linux server:
'`ssh -Y username@login.ifi.uio.no`'
Make sure to request X forwarding by specifying the `-Y` command line option. This would probably be the best option for most **Linux** and **Mac** users, at least. However, it's also possible to connect via ssh from **Windows** — for an example of how to set things up see `http://www.math.umn.edu/systems_guide/putty_xwin32.html`.

2 How to configure Emacs

This is a one-time operation: Once connected to a Linux machine at IFI, copy the contents of '`~erikve/.emacs`' to the '`.emacs`' file in your own home directory. The file contains settings that are read by Emacs at start-up (written in the Emacs Lisp dialect). The file is also available from the INF4820 web page:
`http://www.uio.no/studier/emner/matnat/ifi/INF4820/h12/undervisningsmateriale/dot-emacs.txt`
Ps: If you're unsure about how to copy the contents of the file, just use the following command;
'`cat ~erikve/.emacs >> ~/.emacs`'

3 How to run Emacs and Lisp

First of all; for an overview of basic Emacs commands and key-bindings, please have a look at the Emacs 'cheat sheet' posted at the course page:
`http://www.uio.no/studier/emner/matnat/ifi/INF4820/h12/undervisningsmateriale/emacs.pdf`.

It could be a good idea to always keep this handy for reference while getting started with Emacs/Lisp. (Note that while the Seibel 2005 book uses the SLIME environment, we will instead be using the Allegro Emacs—Lisp interface. Unfortunately, the two environments differ in how they bind commands to keys in Emacs.)

You can start Emacs from a command line / terminal by simply typing;
'`emacs &`'

Then start the integrated Lisp development environment by giving the following command to Emacs;
'`M-x lisp RET`'

(In detail, this means the following: In Emacs, press and hold the 'Meta' / 'Alt' button while pressing 'x'. This will present you with a prompt at the bottom of the Emacs window (the so-called *minibuffer*) where you type the command '`lisp`' and then press 'Enter' / 'Return').

This should start the Lisp process and create an Emacs buffer called '`*common-lisp*`'. Here you can interact with the top-level Lisp interpreter, i.e., type in s-expressions at the prompt, have them evaluated, and the result returned to you. (This is the so-called **REPL**; read-eval-print-loop.) You can also **write and save your Lisp code in a file** (using the extension '.lisp'), and then **load** that file into the Lisp environment. Loading a Lisp source file can be done is several ways: Besides using the menu, you can type '`:ld filename`' at the Lisp prompt, or issue the key-command '`C-c C-b`' in the buffer displaying the file (this tells Emacs to evaluate the buffer). You can then test the functions you've defined in the file by calling them from the Lisp prompt. (If you then go on to make changes in the file you need to re-load it to make the changes visible at the top-level.)

Note, *loading* a file, as described above, typically means using *interpreted* code. Although this has certain benefits when debugging code, *compiled* code tends to run faster. If you want to ensure that you **compile** your code, use '`:cl filename`' rather than `:ld`.

Before quitting Emacs ('`C-x C-c`'), be sure to first type `:exit` at the Lisp prompt to quit Allegro CL.

erik.velldal@ifi.uio.no