CSE341: Programming Languages
Using SML and Emacs
Winter 2013

Contents

1   Overview

For roughly the first half of the course, we will work with the Standard ML programming language, using the SML/NJ (Standard ML of New Jersey) compiler. You will need SML/NJ and a text editor on your computer to do the programming assignments. Any editor that can handle plain text will work, but we ***strongly recommend*** using Emacs and its SML Mode, which provides good syntax highlighting, indentation, and integration with an SML environment. While Emacs does not have the look-and-feel or tool-integration of many modern integrated development environments (IDEs), it is a versatile tool well-known by many computer scientists and software developers.

This document describes how to install, configure, and use Emacs, SML/NJ, and SML-Mode-for-emacs (henceforth SML Mode) on your computer. These instructions should work for recent versions of Windows, Mac OS X, and Linux.

We also describe how to use the machines in the CSE undergraduate department labs, where most of what you need is already installed on both the Windows and Linux machines.

2   Using the Department Labs

If you use the machines in the department's basement labs, then recent versions of Emacs and SML/NJ are alread installed for you, allowing you to skip Sections 3, 5, and 7 below. So here is all you need to do:

1. Open Emacs. On Windows, go to the start menu, search for *emacs* and open the program. On Linux, go to *activities*, then *applications*, then the icon for *emacs*.

2. Read Section 4 to get familiar with Emacs.

3. Follow the instructions in Section 6 to install SML Mode for Emacs. You should need to do this only once.

4. Read Section 8.

If you are new to the department labs, it would also help to read up on the department information on where to save your files, how to behave in the lab, etc. Suggestions for particularly important information to include in this document is welcome — don't be shy.

3   Emacs Installation

(We strongly recommend Emacs version 24.X (for any X) so that you can use the most recent version of SML Mode. Earlier versions of SML Mode are fine, but they are more difficult to install. You can check the version of an Emacs installation in several ways, including the "About Emacs" option under the "Help" menu. Installing version 24 is easy, so we recommend doing so even if you already have an older version.)

Directions depend on your operating system:

Windows:

Download a zip archive of the most recent full version, currently Version 24.2, available at `http://ftp.gnu.org/gnu/emacs/windows/emacs-24.2-bin-i386.zip`. (More information and other versions of Emacs are available at the GNU Emacs website, `http://www.gnu.org/s/emacs/`.)

Unpack the downloaded zip archive file `emacs-24.2-bin-i386.zip` by right-clicking it and choosing *Extract All*. This should produce a folder called `emacs-24.2-bin-i386`. Open it to find a single folder named `emacs-24.2`. Move this `emacs-24.2` wherever you want, but pick a permanent place (i.e., do not move it again after the next step).

Once you have moved the `emacs-24.2` folder to where you want, look inside to find `bin\addpm.exe` (the `.exe` extension might not be visible in the folder window depending on how you have Windows configured). Double click on this file to run it one time and it should add a *Gnu Emacs* folder to your *Start* menu and do some other setup operations. Open the *Start* menu and select *Gnu Emacs → Emacs* (or just type *emacs* in the search box) to launch Emacs.

Mac OS:

Download Emacs as a Mac OS X application from `http://emacsformacosx.com/`. Open the disk image file (*.dmg*) and drag the Emacs application to your Applications folder. If you prefer another version of Emacs, such as the more primitive one on the command line or Aquamacs (`http://aquamacs.org/`), you can use it, but you are more likely to have a version older than Emacs 24.X, which will make it more difficult to install SML Mode.
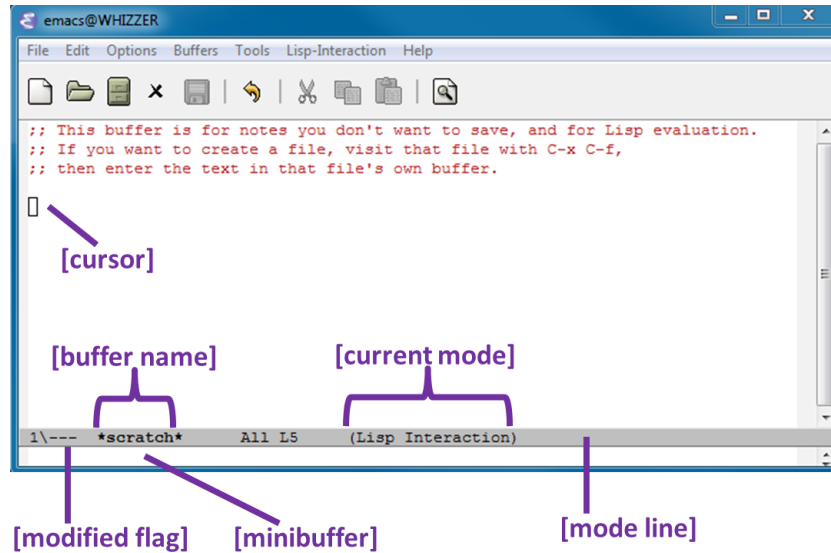
Linux:

Emacs is probably already installed. If not, use your package manager to install it. On Ubuntu and other Debian derivatives, try `sudo apt-get install emacs`. On Fedora, try `sudo yum install emacs`.

4   Emacs Basics

Using Emacs feels a little different than using other editors; it can take some getting used to, especially the keyboard shortcuts. Fortunately, Emacs has buttons and menus to help you adjust if that is your style. The following is a short primer on Emacs terminology and keyboard commands.

Beyond the basics described here, which should be all you need, there are countless free resources available. An introductory "tour" is at `http://www.gnu.org/software/emacs/tour/`. For more information, see the Emacs Reference Manual at `http://www.gnu.org/software/emacs/manual/html_node/emacs/index.html` (also available within Emacs from the Help Menu), the Emacs Wiki at `http://www.emacswiki.org/`, or the Emacs Tutorial (within Emacs from the Help Menu).

```
;; This buffer is for notes you don't want to save, and for Lisp evaluation.
;; If you want to create a file, visit that file with C-x C-f,
;; then enter the text in that file's own buffer.
```

[cursor]

[buffer name]   [current mode]

```
1\---   *scratch*      All L5    (Lisp Interaction)
```

[modified flag]   [minibuffer]   [mode line]

- The cursor is a rectangular block and is referred to as the **point**.

- The **mode line** displays information about the **buffer** displayed in the current **window**. A buffer is a logical "thing" that you are working on. When you open a file, it will be loaded into a buffer, typically with the same name as the file.

- Every buffer is edited in a **mode**. The most basic mode is **Fundamental**, which provides only the most basic Emacs editing features. There are modes for many different programming languages.

- There are many "special" buffers that do not correspond to loaded files. The one above is called ***scratch***. This buffer runs in Lisp Interaction mode, which means that you can interactively type and evaluate expressions in the Lisp programming language. But we won't do that.

Emacs uses many key combinations involving the **Control** and **Meta** keys. Such key combinations are denoted **C-x** (Control-x (lowercase)) or **M-x** (Meta-x). On keyboards that don't have Meta (just about all keyboards today), try *Alt* on PC keyboards or the funny-symbol-with-four-circles or *Option* keys on a Mac keyboard. (Emacs might complain about "Super" if you get the wrong one. If you or Emacs gets confused about what you are trying to type, use **C-g** to cancel your current command and start fresh.) If none of these work, use *Esc*, but when trying to type M-x, for example, type Esc *then* type x. (This is only if using Esc as Meta. The other Meta "substitutes" work as usual: hold while pressing the second key.)

A sequence of key presses is written like **C-a C-b M-x**, which would mean do the three actions in sequence: Control and lower-case a, then Control and lower-case b, then Meta and lower-case x.

The most important commands in Emacs:

- **C-x C-c**: Quit Emacs

- **C-g**: Cancel the current action

- **C-x C-f**: Open a file (whether or not it already exists)

- **C-x C-s**: Save a file

- **C-x C-w**: Write a file (probably more familiar to you as *Save as...*)

Cut, copy, paste:

- Highlight text with the mouse or by hitting **C-*Space*** to *set a mark* and then moving the cursor to highlight a region.

- **C-w**: Cut a highlighted region

- **M-w**: Copy a highlighted region

- **C-k**: Cut (*k*ill) from the cursor to the end of the line

- **C-y**: Paste (*y*ank)

Some other useful commands:

- **C-x 2**: Split the window into 2 buffers, one above the other (Use the mouse or **C-x o** to switch between them)

- **C-x 0**: Undo window-splitting so there is only 1 buffer

- **C-x b**: Switch to another buffer by entering its name

- **C-x C-b**: See a list of all current buffers

Getting help within Emacs: In addition to the help button/menu on the right...

- **C-h**: Help. Hitting this will display a short message in the minibuffer: `C-h (Type ?  for further options)`.

- **C-h b**: Key *b*indings. This lists all key bindings that are valid for the current mode. Note that key bindings change from mode to mode.

- **C-h a**: Command *a*propos. After typing `C-h a` you can type a symbol and a window will appear that lists all symbols and functions that match that phrase.

More advanced Emacs hacks (optional):

If you are curious, try some of these once you have finished the rest of the setup instructions. They are unnecessary for any of the work we will do, but may be convenient.

- Change the colors of your syntax highlighting. In version 24 of Emacs, M-x customize-themes is a good place to start.

- General customization interface: Open the *Options* menu and choose the first item under *Customize Emacs*. This will let you customize Emacs through a sort-of-graphical interface. It saves all your settings in a file in your "home" directory, `~/.emacs`.

- Much, much, much more: Emacs calls itself an *extensible* editor for a reason.

5 SML/NJ Installation

Directions *first* depend on your operating system, but then see "All Systems: Check your SML Installation" below.

Windows:

Download and run the `smlnj.msi` installer available at `http://www.smlnj.org/dist/working/110.75/`. This will add an item for *SML of New Jersey* to your *Start* menu and add a command `sml` that you can use at the command line.

Mac OS:

Download the installer available at `http://www.smlnj.org/dist/working/110.75/`. Notice there are two different *.dmg* files, so be sure to get the correct one for your operating-system version. Open the disk image file *.dmg* and run the installer.

Once the installation is complete, use Emacs or another text editor to edit the file `.bash_profile` in your home folder. (In Emacs you can do this via: `C-x C-f ~/.bash_profile`, notice the three characters "tilde, slash, dot.") If the file does not already exist, create it. Add the following line to make sure SML is visible on the command line (if you chose a custom install location, change /usr/local/smlnj-110.75 to whatever that location is):

```
export PATH="$PATH:/usr/local/smlnj-110.75/bin"
```

This tells your shell (the program that you interact with in the terminal) to add the SML/NJ directory to the paths it searches to find programs. (If you are not using the bash shell, which Mac OS X has used by default since 10.3, the syntax will be different.)

Finally, you will need to run your `.bash_profile` to deploy the changes you have made into your environment for the present session. To do this, run:

```
source .bash_profile
```

You need to do this only once — afterwards, each new terminal that you open will automatically run the `.bash_profile` for you.

Linux:

If your package manager has a package for SML/NJ, install it. Otherwise, follow the "Unix" instructions at `http://www.smlnj.org/dist/working/110.75/`.

All Systems: Check your SML Installation

1. Open a terminal window and type `sml` followed by *Enter/Return.* To open a terminal window:

   - Windows: *Start → All Programs → Accessories → Command Prompt*, **or** Windows 7 just use the Start Menu to search for the cmd.exe program and run it.
   - Mac OS: Open *Applications/Utilities/Terminal.app*.
   - Linux: Various ways: any shell should be fine.

2. You should see a prompt that looks like this:

   ```
   Standard ML of New Jersey v110.75 [built: ...]
   ```

3. Make sure everything is working by typing a very simple SML program at the prompt:

```
1 + 1;
```

4. Hit *Enter/Return.* In response, the SML interpreter should print something like this:

```
val it = 2 : int
```

5. To exit the interpreter, type Control-Z and then Return on Windows and Control-D on Mac or Linux.

6   SML Mode for Emacs Installation

SML Mode is an extension to Emacs that is not Emacs itself or SML/NJ itself. It displays SML code nicely with syntax coloring and clean indentation, and provides a way to run SML from within Emacs.[1]

To install the current version of SML Mode (currently 6.2), follow these instructions from within Emacs:

- Run the command `M-x list-packages` (and then *Return/Enter*). If the `list-packages` command does not exist, your Emacs version is too old. You can upgrade Emacs or follow more difficult SML Mode installation instructions below.

- Find `sml-mode` and click on it with your mouse.

- Click on `install` with your mouse.

- Exit and restart Emacs.

- Read below to see if you need to follow a couple more steps (more likely under Mac and Linux).

To verify that SML Mode is properly installed, let us check that it does indentation/coloring for SML files and that you can create the SML read-eval-print-loop (REPL) from within Emacs.

First, edit an existing or new SML file (try `C-x C-f test.sml` to create a new file if nothing else is handy). You should see the mode display at the bottom of the Emacs window change from Fundamental (or whatever it was) to SML. If you enter a line of code like `val n = 1;` you should see colors highlighting the keywords and variable names. When you are editing code, whenever you hit the *Tab* key, Emacs will try to reindent the current line appropriately.

Second, while the cursor is in an SML buffer (i.e., you are editing an SML file), run `C-c C-s`. This should split the window and create an SML prompt in a new buffer. In that buffer, you should be able to type `1+1;` at the prompt and see 2 as the result.

If you are seeing syntax highlighting, but the `C-c C-s` command fails with an error message, Emacs is probably having trouble finding the SML program. You can hopefully fix this as follows:

Mac OS: In Emacs, edit your `.emacs` file by `C-x C-f ~/.emacs` (that is tilde, slash, dot, emacs) to open the file. Paste in these lines:

```
(setenv "PATH" (concat "/usr/local/smlnj-110.75/bin:" (getenv "PATH")))
(setq exec-path (cons "/usr/local/smlnj-110.75/bin"  exec-path))
```

Save the file (`C-x C-s`). Exit and restart Emacs.

Linux: Find where `smlnj-110.75` was installed. Then follow the Mac OS instructions above, but replacing `/usr/local` with the appropriate path.

---

[1]Thanks to Stefan Monnier for maintaining SML Mode. The website is `http://www.iro.umontreal.ca/~monnier/elisp/`, but you do *not* need to go there to install SML Mode.

7 Manual SML Mode Installation for Older Emacs Versions

If you are using Emacs 24.X, you should be able to use the previous section and **skip this section entirely, so skip to the next section**. These instructions are only for people who want to use an older version of Emacs, which requires using an older version of SML Mode and following different installation instructions.

Go to `http://www.iro.umontreal.ca/~monnier/elisp/` and download version 5.0 (see link "Download 5.0" or just download directly (`http://www.iro.umontreal.ca/~monnier/elisp/sml-mode-5.0.tar.gz`). Unpack it:

- Windows: You need some program that can handle a **.tar.gz** file. The Cygwin tools (`www.cygwin.com`) suffice, or an easier-to-use program is 7-Zip (`www.7-zip.org/`). For 7-zip, install it. Then right-click the .tar.gz file and choose *7-Zip → Extract Here*. If this creates a .tar file instead of a folder, do *7-Zip → Extract Here* on that file as well.

- Mac OS: double-click the .tar.gz file.

- Linux: Either:
  - When downloading, choose *Open with [Archive Manager]* if given the option. Otherwise, save the file and double-click it, which should open it in Archive Manager. Drag the *sml-mode-5.0* folder to the Desktop or wherever you want to store it.
  - Save the file. In a terminal, `cd` into the directory where you saved it and run `tar -xfz sml-mode.tar.gz`.

You should get a folder named `sml-mode-5.0`. You can place this folder anywhere you like.

To get SML Mode to work with Emacs, you need to create/edit a `.emacs` file in your "home" directory/folder that Emacs searches when it starts up. Open this file in Emacs via the command `C-x C-f ~/.emacs`.

Add the following line to the `.emacs` file:

```
(load "/path/to/where/you/stored/sml-mode-5.0/sml-mode-startup")
```

but replacing `path/to/where/you/stored` with the appropriate path:

- Windows: Hold down the *Shift* key as you right-click the `sml-mode-5.0` folder and choose *Copy As Path*. Paste this into Emacs. Then replace each \ with / because \ is a special character in Emacs settings files. And add `/sml-mode-startup` at the end. So you might end up with something like `Z:/look/here/sml-mode-startup`.

- Mac OS: Select the folder in the Finder and choose *File → Get Info*. Copy the *Where:* field and add `/sml-mode-startup`. Now also add the following to your `.emacs` file in addition to the line above:

```
(setenv "PATH" (concat "/usr/local/smlnj-110.75/bin:" (getenv "PATH")))
(setq exec-path (cons "/usr/local/smlnj-110.75/bin"  exec-path))
```

- Linux: If SML Mode seems to work, but you cannot get the SML REPL to run in Emacs, then find where `smlnj-110.75` was installed and add the two lines under the Mac OS instructions above, but replacing `/usr/local` with the appropriate path.

(There are much more elaborate installation instructions in the documentation included in the sml-mode-5.0 folder, but these are aimed at system administrators who might want to install SML Mode for an entire site instead of a single user. You do not need these.)

Now verify that SML Mode is properly installed, following the instructions in the previous section.

Finally, if you are in SML Mode but your SML code does not have syntax coloring, add this line to your `~/.emacs` file and then restart Emacs:

```
(global-font-lock-mode t)
```

This should be necessary only if your Emacs version is older than 23.X.

8 Using the SML/NJ REPL (Read-Eval-Print Loop) in Emacs

At this point, we are done installing! This section shows you how to run SML programs from within Emacs. It assumes you already have an SML file or can write your own SML program in a new one.

1. Edit a file with extension `.sml`. You should be in SML-mode, using *Tab* to indent your code well.

2. To create the `*sml*` buffer (which holds the REPL), type `C-c C-s` (and then *Return/Enter*) in the buffer with the `.sml` file. (Note: This will not work in the `*scratch*` buffer that Emacs starts in because this buffer is not in SML Mode.)

3. Keep the `.sml` file(s) you are working with for a particular assignment in the same folder. When you type `C-c C-s` to start the REPL from a buffer for `foo.sml`, the REPL will look in the right folder for `foo.sml` when you type `use "foo.sml"` and will look in the same folder for any other file you use such as `foo_tests.sml`. This is less confusing than trying to keep track of different folders and paths while using the REPL although that is possible.

4. To end and restart a REPL session, type `C-d` (to end it) and `C-c C-s` (and then *Return/Enter*) (to restart it). You must type `C-d` while in the `*sml*` buffer; you can type `C-c C-s` from the `*sml*` buffer or a buffer with a `.sml` file.

5. By ending and restarting a session, the new session has an empty environment. Your earlier interactions are still in the `*sml*` buffer, so you can save them, cut-paste them, etc., but they have no effect on the evaluation in the restarted REPL session.

6. Evaluation can go into an infinite loop.
   - This has likely occurred if you are not getting the "- -" prompt back and nothing appears to be happening.
   - `C-c C-c` will interrupt evaluation and get you your prompt back.

7. If you forget to end your binding with a ";" character, the REPL will print an "=" character on the next line, which is just its way of saying, "you are not done – continue your binding," so type a ";" and hit *Return/Enter*. This is not an infinite loop (nothing is being evaluated; the REPL is waiting for you) so `C-c C-c` does not do anything.

8. If the printed result looks "pretty good," but part of what you expected to see has been replaced by a "#" or "...," do not worry. The REPL has a limit on how many characters it prints, which is good since you might make a large value, such as a list with tens of thousands of elements. You can adjust the limit if you want.

Advice You Will Wish You Followed!

In each REPL session, follow this pattern:

- First type `use "foo.sml";` for any SML files you want to use.

- Then use the REPL manually as long as you wish.

- After using the REPL to test something, do *not* use `use` to load (or reload) any more files.

- When tempted to violate the previous point, end and restart your REPL session before continuing.

Why: `use "foo.sml"` has a very simple semantics: it adds the bindings in the file to the environment in order. These may or may not shadow bindings from the last time you typed `use "foo.sml"`, depending on how `foo.sml` changed. This confuses even expert programmers until they train themselves to follow the pattern above.

If you find yourself typing the same non-trivial things repeatedly in the REPL, stop wasting your time.

- Move the repeated parts to a second file, e.g., `test.sml`.

- Then, when you restart your session, begin with `use "foo.sml"; use "test.sml";`.

- In fact, there is an even faster way:

  - Begin `test.sml` with the expression `use "foo.sml";`
  - Then begin your session with `use "test.sml";`

Note: Do *not* put `use "foo.sml"` in `test.sml` *and* begin your session with `use "foo.sml"; use "test.sml";`. That will evaluate the bindings in `foo.sml` twice, which is confusing.

If you develop some emotional attachment to the transcript of your `*sml*` buffer, you can save it to a file just like any other buffer. But after you do, it's not an `*sml*` buffer anymore, so you will have to create a new `*sml*` buffer from a buffer in SML Mode via `C-c C-s`.