Software and tools

Mark Armstrong

September 2, 2020

Contents

1 Introduction

This document discusses the software and tools needed for CS3mi3, Principles of Programming Languages, for the fall 2020 session.

1.1 IMPORTANT: Your responsibilities

It is not the responsibility of the instructor or the teaching assistants to install the required software and tools on your system.

We will provide docker images for testing submissions, along with basic usage instructions. Installing and learning the basics of Docker is your own responsibility.
We recommend installation of the programming environments and tools locally on your own system for your convenience while working on the assignments. This setup is your own responsibility.

2 Docker

We will use Docker to ensure a common testing environment for all students and staff.

Docker images will be distributed which setup the required language tools for each assignment and include some sanity-checking unit tests for your code.

Similar images with more complete unit tests will make up part of the marking process of assignments.

2.1 TODO Getting started with Docker

3 Editors

Any text editor should be adequate for use in this course. I recommend against "full-fledged IDEs"; it is likely some languages we use will not be supported by any choice of IDE.

I do recommend Emacs as a text editor, though warn that its use involves a steep learning curve and likely a long configuration time —most likely an unending configuration time. The Spacemacs and Doom (Emacs) configurations alleviate at least some of the setup time; they are preconfigured with many commonly used features and support for many programming languages.

Atom, VS Code and Sublime Text are other extensible text editors, any of which is also an excellent choice.

4 Version control

We make frequent use of Git throughout this course for version control.

We will make use of the course GitHub repository and its github.io website for course content distribution.

A repository on the CAS departmental Gitlab server will be created for each student for course work submission, as your accounts there are tied to your macid.

5 Online communication

Online communications, **including for this term all lectures and tutorials**, will be conducted using Microsoft Teams.

By the beginning of the course, you should have access to a Team for the course when logging in using your McMaster email and MacID password.

6 Programming languages and their implementations

We plan to use four programming languages for homework and assignments throughout the course. Each language will require an implementation to be installed.

This section provides information on those implementations, as well as some additional resources for each language.

6.1 Scala functional:objectoriented

A Java-descended, *pure* object oriented language with full functional programming support on the JVM.

- The Scala homepage.
- Hands-on Scala Programming, a very recent book with the first few chapters available freely online.

6.2 Prolog logical

The prevalent logical programming language. Program by *describing the problem*, rather than providing an algorithms.

6.3 Ruby objectoriented:scripting

A *nearly pure* object-oriented *scripting* language featuring extremely flexible syntax.

6.4 Clojure functional:LISP

A Lisp (LISt Programming) functional language on the JVM.