Syllabus

Eric Mayhew & Jacob Errington*

Winter 2018

Website: `http://computing-workshop.com/`

Location: B21, 651 rue Sherbrooke Ouest (Northeast corner of University street)

Time: From 1:30 PM to 3:00 PM on the following days:

- January 26,
- February 2, 9, 16, 23,
- March 2, 16, 23, 30.

After each session there is a one-hour homework tutorial during which the facilitators will give hands-on help with whatever homework was assigned during the session.

Goal: At the end of the workshop, particpants will have created their own static site by themselves. Participants will have a deep understanding of their computer, from the circuitry to the operating system.

Description

Computing Workshop is a 9-session workshop designed to help people understand how computers and websites work. This workshop will cover hardware, software, and web development to individuals with little to no background in computers using cooperative, problem-based learning methods. Each class is 1.5 hours long and has an estimated 1 hour worth of homework that accompanies each lesson.

Rationale

We created this workshop to provide people with the support to feel comfortable using a computer. Technology is pervasive in our society, but rarely are individuals provided with meaningful education in how to use it. Our workshop looks to activate participant's motivation by providing them with hands on, interactive lessons covering fundamental knowledge with respect to computers. This workshop uses a co-constructivist teaching approach to help our participants develop their own tools and understanding of computers.

*Special thanks to Building21 (McGill's Office of Student Life and Learning) and to Anita Parmar. Computing Workshop would not have been possible without her tremendous support!

Lesson sequence

0. Intro, first glance at hardware and abstraction
1. Binary, circuits, and intro to Haskell
2. Processors, Haskell types and functions
3. Memory, virutal machine, Haskell functions continued
4. What happens when you turn a computer on, OSs, Haskell functors
5. Hardware wrap up, intro to github, Haskell monads
6. Designing your website, intro to web development, hakyll
7. Creating website components, computer networking, working on your website
8. Finishing your website, online community, conclusion

Learning goals

Computers

- identify key hardware components and their function;
- identify core services provided by an operating system;
- identify the boundaries and relationships between application, operating system, and hardware;
- explain the process of downloading a resource from a remote host;
- explain the role of HTML, CSS, and JavaScript in displaying a web page in a browser;
- categorize different networking protocols (application vs. hardware);
- break down the process from computer power-on to operating system startup;
- interact with online communities and tools;
- appreciate open source material;
- navigate online resources and take autonomy of their own learning.

Programming

- Translate a word problem into code: identify key types and dataflow;
- use Haskell libraries effectively;
- differentiate between values, types, types classes, and kinds;
- type check expressions;
- implement and use higher-order functions and partial applications;
- use the `IO` monad to create programs with side-effects;
- understand general patterns (e.g. `Functor`) when applicable and use them;
- use `do`-notation for a variety of monads: `Maybe`, `Either a`, …;

Web development

- Design and implement a website;
- apply appropriate aesthetics to the design of websites: golden ratio, negative space, color theory, …;
- use appropriate elements of HTML;
- make responsive websites;
- build reusable web components;
- use appropriate web development tooling (e.g. static site generators).