CPS506 - Comparative Programming Languages
Introduction

Dave Mason - Alex Ufkes
Department of Computer Science
Ryerson University

RYERSON UNIVERSITY

Who is Dave?

Dave Mason

Who is Alex?

Alex Ufkes

What is the Course?

https://cps506.cs.ryerson.ca

Responsibilities

- Our responsibilities:
  - be fair, evaluate, uphold standards
  - curate
  - tell you what we know - in the best way we know how to do
  - try to inspire you
- Your responsibilities:
  - **learn!!**
  - be fair, don't cheat
  - oh, and did I mention **learn** - by listening, by interacting, by reading, by coding

Radical Honesty

- I wish you all would earn As
- I'll let you in on a secret, getting an A is not hard
- - do assignments & labs, read reference material, attend class, ask questions until you have an answer that you understand, decide to treat the course as something valuable to you and have fun
- I believe you all can at least earn Bs
- Based on past experience, I expect a dozen or so will fail the course
- I appreciate those of you who will take the advice above - helping you learn is a joy
- I resent those of you who will give a half-hearted effort, and blame me for you not learning
- I have some anxiety about the course, as there are always problems with courses, and I don't want you to have a negative experience

Version Control

- Efficiently maintain every step in the development process
- Excellent for individual development
- **Essential** for team development
- *Every* successful commercial organization uses one (or more)
- RCS, SCCS, CVS - older, non-distributed
- Git, Mercurial, Fossil, Arch, Bazaar - distributed, open-source

Use Cases

- lets you refactor and experiment freely
- support branches for support/development
- merge changes from others
- binary search for regressions
- some people commit daily
- some people commit after every green test
- works with Continuous Integration (Jenkins, Nix)

Git

- very popular DVCS - GitHub, GitLab
- designed to support Linux Kernel development - replace Bitkeeper
- oriented to very large projects with thousands of contributors
- many features to support that model of development
- arcane features easy to misuse

Fossil

- very convenient DVCS
- designed to support SQLite development
- many features to support smaller team development
- simpler interface - avoid forks, maintain consistency
- single binary on each platform - easy installation, self-hosting
- includes web GUI, Wiki, bug tracker

Fossil for CPS506

- upload SSH key - using `submit-cps506`
- automatically creates a `.fossil` for you

You've seen the Imperative Paradigm

- C (& assembler) are classic imperative
- everything done by changing values of variables
- fine for small programs
- difficult to scale up - code size and parallelism
- programmer must model state
- - in their head!
- two widely used paradigms to address these problems: OO and functional

Definitions

- Syntax - the externally visible representation of a program
- Semantics - the meaning of a program
- Pragmatics - non-functional characteristics of a program (environment)
- Implementation - a particular set of pragmatics to make a program executable

Syntax

- Simplicity - how much to learn
  - size of the grammar
  - complexity of navigating modules/classes
  - complexity of the type system
- Orthogonality - how hard to learn, how do features interact
  - number of special syntax forms
  - number of special datatypes
  - type system
- Extensibility - how can language align with problem
  - functional
  - syntactically
  - defining literals
  - overloading