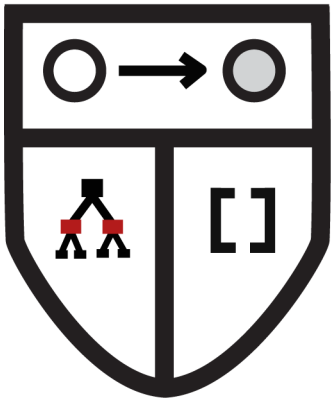


CS 3110

Introduction to 3110

Nate Foster

Spring 2018

Welcome!

- Programming isn't hard
- Programming well is very hard
  - High variance among professionals' productivity: 10x or more
  - Studying functional programming will make you a better programmer, but it requires an open mind

What is a functional language?

A functional language:

- defines computations as mathematical functions
- avoids mutable state

State: the information maintained by a computation

Mutable: can be changed (antonym: *immutable*)

Functional vs. imperative

Functional languages:

- Higher level of abstraction
- Easier to develop robust software
- Immutable state: easier to reason about software

Imperative languages:

- Lower level of abstraction
- Harder to develop robust software
- Mutable state: harder to reason about software

You don't have to believe me now.
If you master a functional language, you will. ☺

Imperative programming

Commands specify how to compute by destructively changing state:

```
x = x+1;
a[i] = 42;
p.next = p.next.next;
```

Functions/methods have **side effects**:

```
int wheels(Vehicle v) {
  v.size++;
  return v.numWheels;
}
```

Mutability

The fantasy of mutability:

- It's easy to reason about!
- The machine does this, then that...

The reality of mutability:

- Machines are good at complicated manipulation of state
- Humans are not good at understanding it!
- Mutability breaks *referential transparency*: ability to replace an expression with its value without affecting the rest of the computation

...mutability is not well-suited to writing correct code!

Functional programming

Expressions specify what to compute

- Variables never change value
- Functions never have side effects

The reality of immutability:

- No need to think about state
- Powerful ways to build correct programs

Why study functional programming?

1. Functional languages teach you that programming transcends programming in a language (assuming you you have only programmed in imperative languages)
2. Functional languages predict the future
3. (Functional languages are *sometimes* used in industry)
4. Functional languages are elegant

Why study functional programming?

1. Functional languages teach you that programming transcends programming in a language (assuming you you have only programmed in imperative languages)
2. Functional languages predict the future
3. (Functional languages are *sometimes* used in industry)
4. Functional languages are elegant

Analogy: studying a foreign language

- Learn about another culture; incorporate aspects into your own life
- Shed preconceptions and prejudices about others
- Understand your native language better

Alan J. Perlis

1922-1990

"A language that doesn't affect the way you think about programming is not worth knowing."

First recipient of the Turing Award

for his "influence in the area of advanced programming techniques and compiler construction"

Why study functional programming?

1. Functional languages teach you that programming transcends programming in a language (assuming you you have only programmed in imperative languages)
2. Functional languages predict the future
3. (Functional languages are *sometimes* used in industry)
4. Functional languages are elegant

Functional languages predict the future

- Garbage collection
  Java [1995], LISP [1958]
- Generics
  Java 5 [2004], ML [1990]
- Higher-order functions
  C#3.0 [2007], Java 8 [2014], LISP [1958]
- Type inference
  C++11 [2011], Java 7 [2011] and 8, ML [1990]
- **What's next?**

Why study functional programming?

1. Functional languages teach you that programming transcends programming in a language (assuming you you have only programmed in imperative languages)
2. Functional languages predict the future
3. (Functional languages are *sometimes* used in industry)
4. Functional languages are elegant

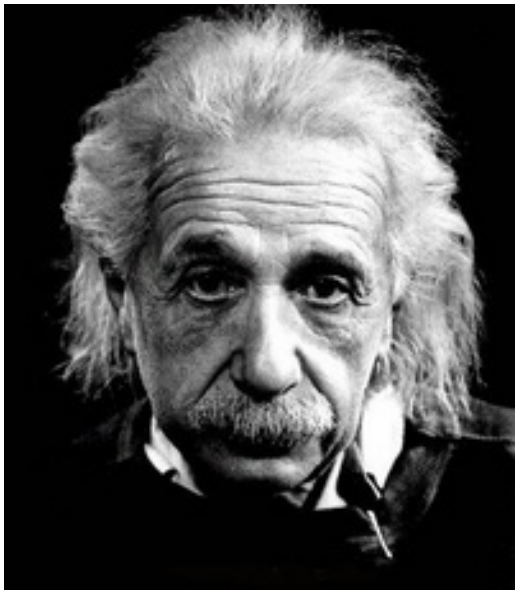
Functional languages in the real world

- Java 8
- F#, C# 3.0, LINQ
- Scala
- Haskell
- Erlang
- OCaml

https://ocaml.org/learn/companies.html

...but Cornell CS (et al.) require functional programming for your *education*, not to get you a job

Albert Einstein



1879-1955

"Education is what remains after one has forgotten everything one learned in school."

Why study functional programming?

1. Functional languages teach you that programming transcends programming in a language (assuming you you have only programmed in imperative languages)
2. Functional languages predict the future
3. (Functional languages are *sometimes* used in industry)
4. Functional languages are elegant

Elegant

Neat
Stylish
Dignified
Refined
Simple
Graceful
Effective
Precise
Consistent
Tasteful

Elegant

Neat Stylish

Beautiful

Precise Consistent

Tasteful

Do aesthetics matter?

YES!

Who reads code?

- Machines
- Humans

- Elegant code is easier to read and maintain
- Elegant code might (not) be easier to write

OCaml

A pretty good language for writing beautiful programs

O = Objective, Caml=not important
ML is a family of languages; originally the “meta-language” for a tool

OCaml is awesome because of...

- Immutable programming
  - Variable's values cannot destructively be changed; makes reasoning about program easier!
- Algebraic datatypes and pattern matching
  - Makes definition and manipulation of complex data structures easy to express
- First-class functions
  - Functions can be passed around like ordinary values
- Static type-checking
  - Reduce number of run-time errors
- Automatic type inference
  - No burden to write down types of every single variable
- Parametric polymorphism
  - Enables construction of abstractions that work across many data types
- Garbage collection
  - Automated memory management eliminates many run-time errors
- Modules
  - Advanced system for structuring large systems

A GLIMPSE OF OCAML...

Languages are tools

Languages are tools

- There's no universally perfect tool
- There's no universally perfect language
- **OCaml is good for this course** because:
  - good mix of functional & imperative features
  - relatively easy to reason about meaning of programs
- **But OCaml isn't perfect** (see above)
  - there will be features you miss from language X
  - there will be annoyances based on your expectations
  - keep an open mind, try to have fun

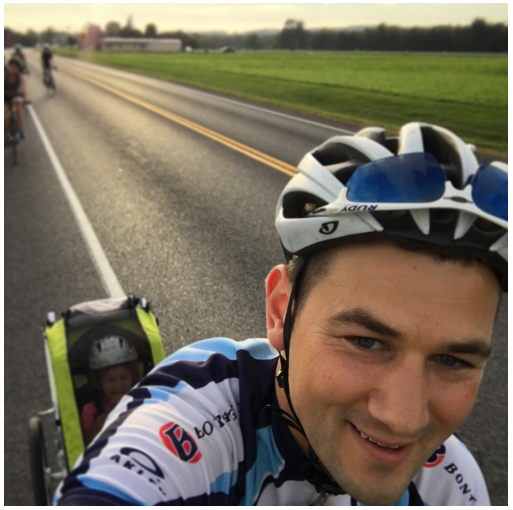
LOGISTICS

Course website

http://cs3110.org

- Full syllabus (required reading)
- Lecture slides & notes
  - Sometimes posted night before, but if not, it's because I'm still tweaking them
  - Supplement, do not replace, attendance
- Labs: practical problems for you to study and talk about in recitations

Course staff



Instructor: Nate Foster

- PhD 2009 UPenn
- Research: programming languages & networking
- Call me "Nate" in this course, or "Dr. Foster" if you're not into the whole brevity thing.

TAs and consultants: 35 at last count

- Grad TAs: Alan Cheng, Dietrich Geisler, Harrison Goldstein, Richa Despanda
- Head Consultant: Irene Yoon

Course meetings

- **Lectures:** TR 10:10-11:00 am
  - Attendance is semi-mandatory, measured by i>clicker
  - Go buy an i>clicker and bring it on Tuesday
  - Laptop use only in designated area

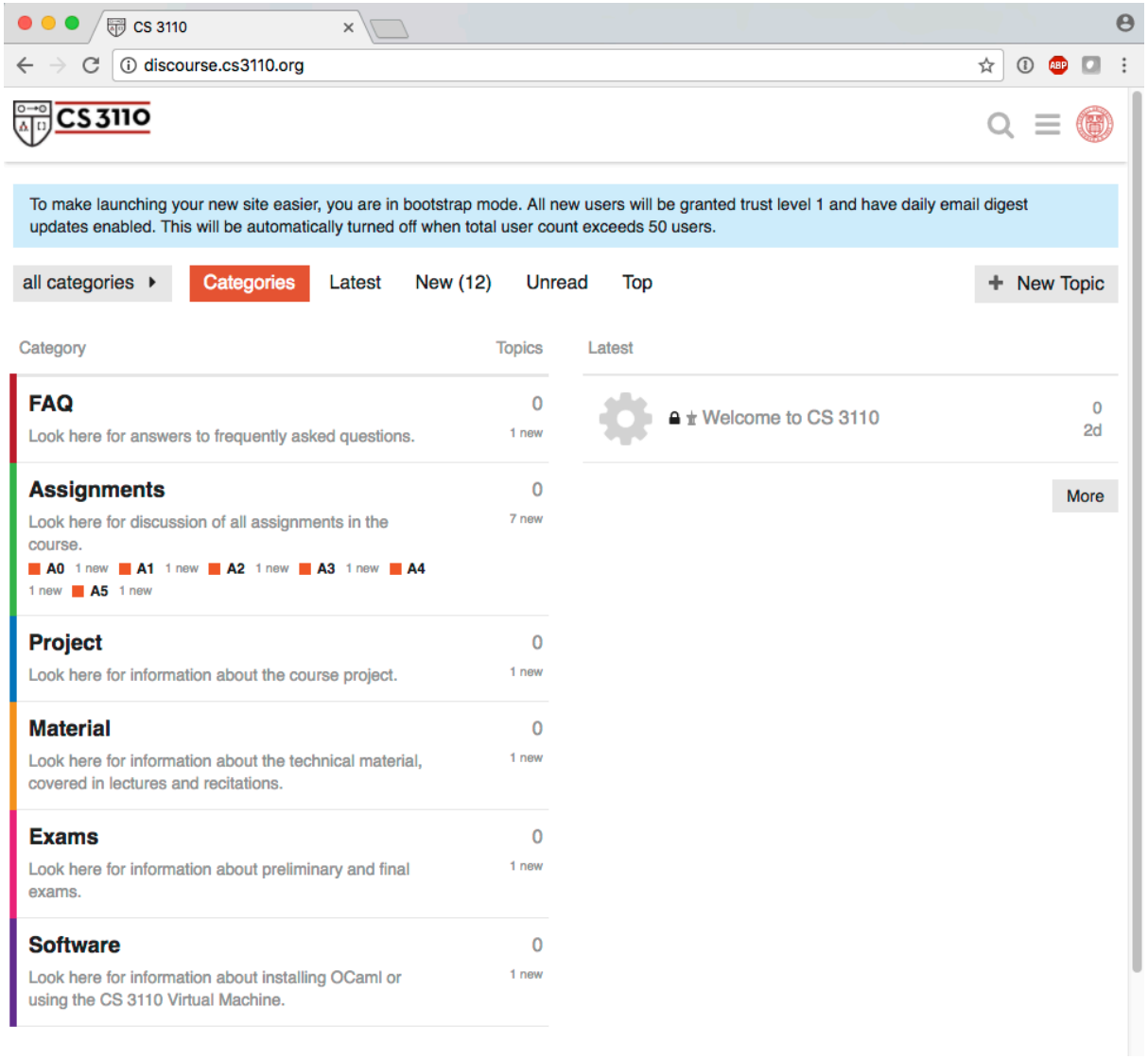
- **Recitations:** mostly MW
  - Attendance is semi-mandatory, measured by TAs
  - TR sections are effectively MW delayed one day
  - You need to attend your registered section
  - Laptops encouraged for working problems

- **Office hours:** coverage nearly every day

Piazza

Last fall: 2700+ posts

http://discourse.cs3110.org



Discourse

Requests from me to you:

- Take questions of broad interest to Discourse or to office hours
- Take questions of interest only to you to office hours
- Treat Discourse as virtual study group, not instructor hotline
- Answer one another's questions: bonus!
- Be brave: eschew anonymity
- Read webpage on "asking technical questions"

Email

- To contact me, send email to cs3110-profs-L@cornell.edu
  - If it's something we can handle in person after the next lecture, I probably won't reply
  - Assume replies may take up to a day

Academic integrity

- You are bound by the Cornell Code of Academic Integrity, the CS Department Code of Academic Integrity, and the CS 1110 Explanation of Academic Integrity
  - All linked from course website
  - You are responsible for understanding them
- I use MOSS to detect plagiarism; it works
- If you have a question about what is or is not allowed, please ask
- If you fear you have committed a violation, tell me before grading commences

Your TODO list for tonight

1. read material on website linked for today: syllabus, tips for success, asking technical questions
2. buy an i>clicker
3. login to CMS
4. login to Piazza
5. install software: instructions linked on website
6. try doing the lab linked for today; bring any questions you have to recitation tomorrow

Upcoming events

- [Monday pm] Drop by my office in the afternoon if you need something immediately
- [today] Office hours start; check calendar on course website for times; consulting hours always in Rhodes 405
- [Monday] Recitations begin (none today); bring laptop
- [Tuesday am] bring i>clicker

…why are you still here? Get to work! ☺