



Class 1: Introduction

cs4414 Spring 2014

rust-class.org

University of Virginia

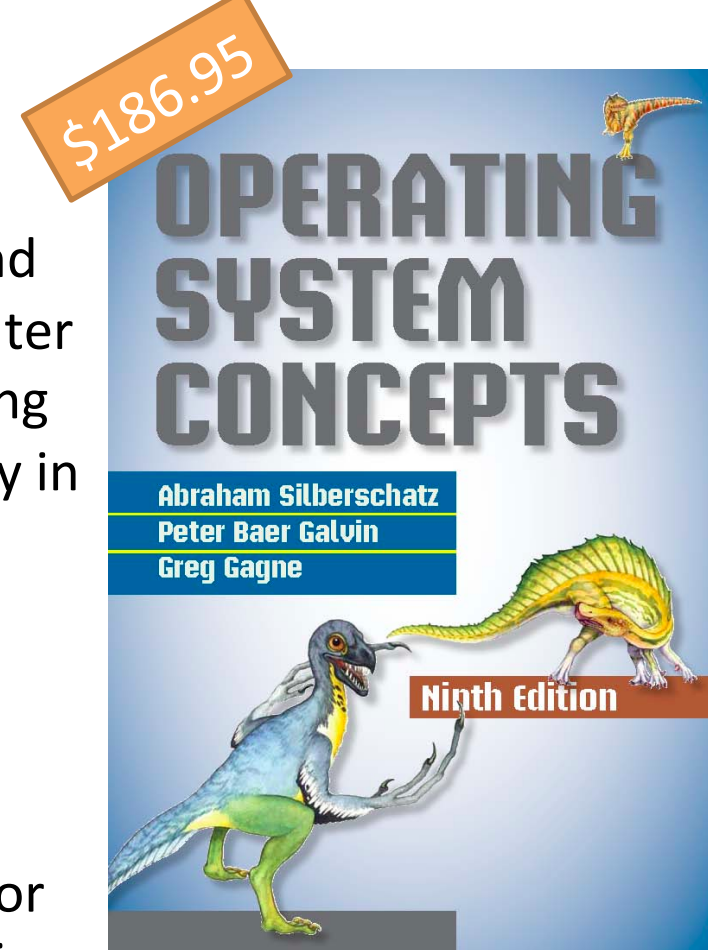
David Evans

Plan for Today

- What is an Operating System?
- Course Overview
- Introducing Rust

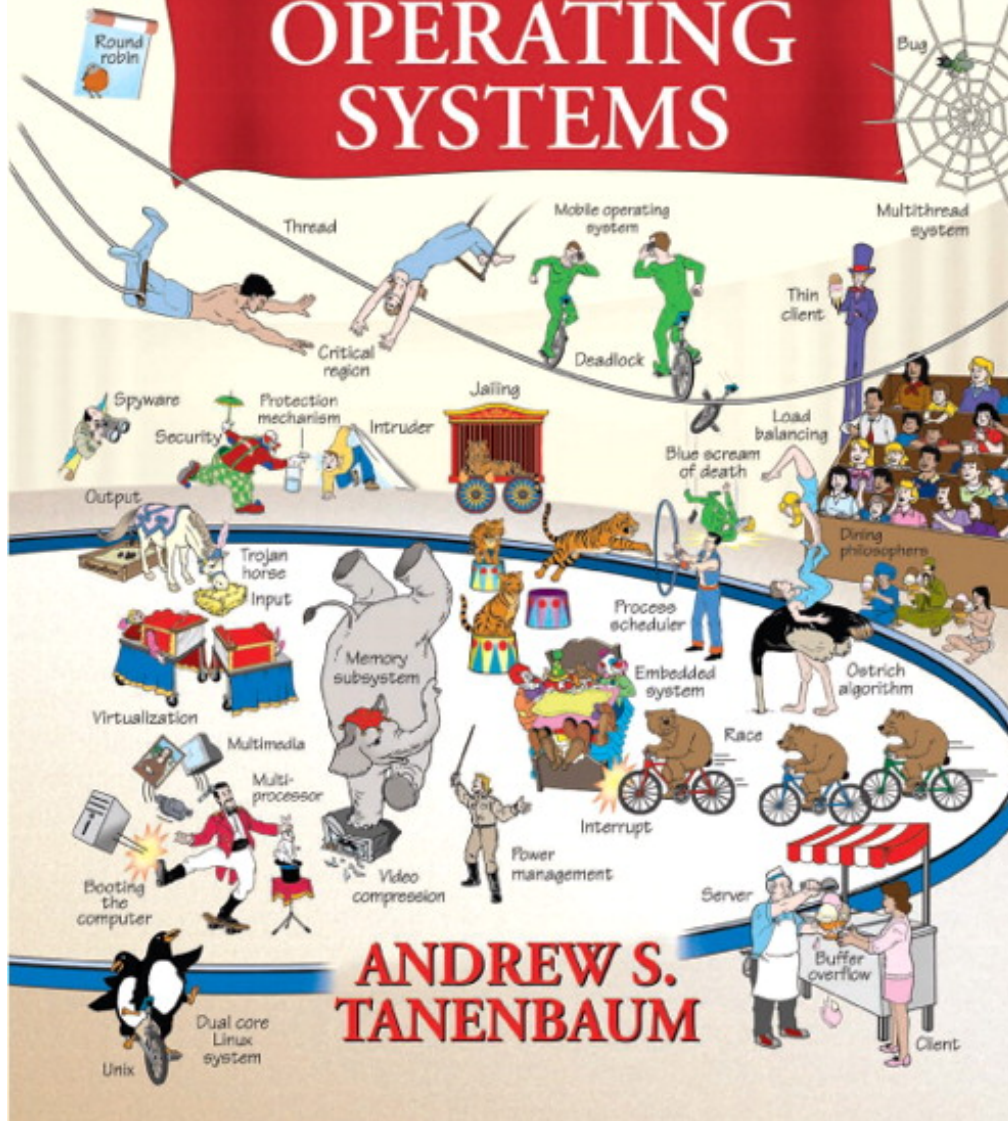
What is an Operating System?

An **operating system** is a program that manages a computer's hardware. It also provides a basis for application programs and acts as an intermediary between the computer user and the computer hardware. An amazing aspect of operating systems is how they vary in accomplishing these tasks. Mainframe operating systems are designed primarily to optimize utilization of hardware. Personal computer (PC) operating systems support complex games, business applications, and everything in between. Operating systems for mobile computers provide an environment in which a user can easily interface with the computer to execute programs. Thus, some operating systems are designed to be **convenient**, others to be **efficient**, and others to be some combination of the two.



\$186.00

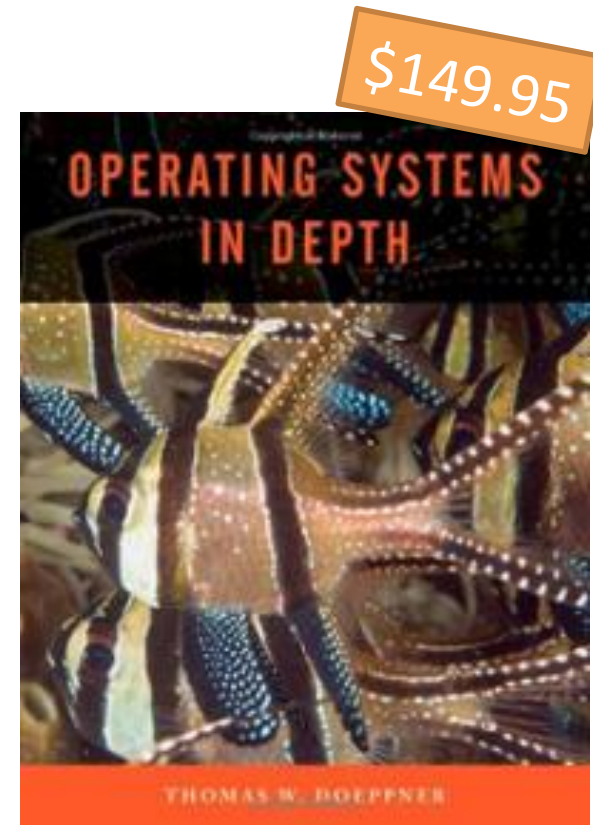
MODERN^{3e} OPERATING SYSTEMS



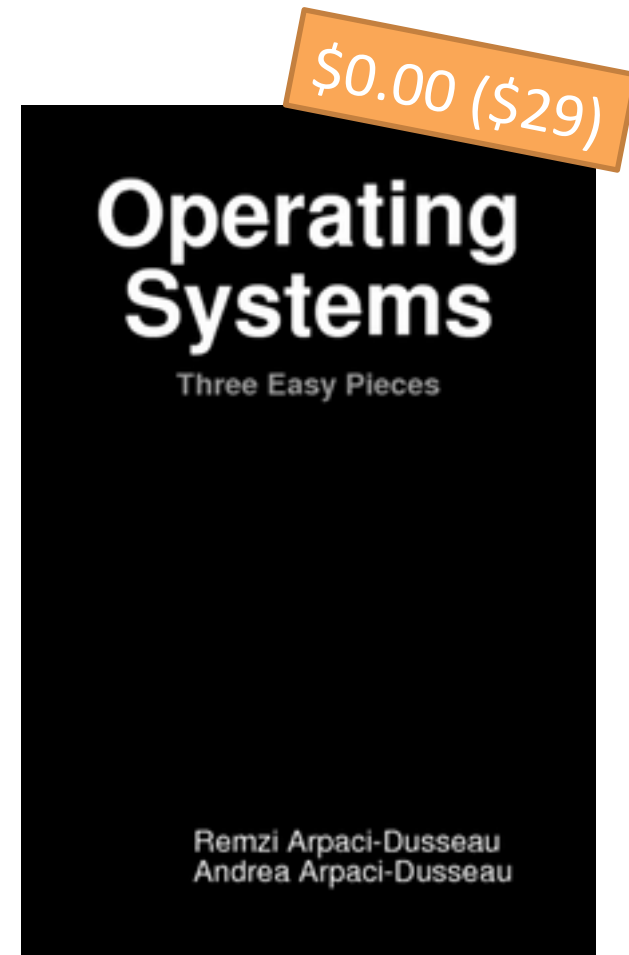
It is hard to pin down what an operating system is other than saying it is the **software that runs in kernel mode** – and even that is not always true. Part of the problem is that operating systems perform two basically unrelated functions: providing application programmers (and application programs, naturally) a clean abstract set of resources instead of the messy hardware ones and managing these hardware resources.

What's an operating system? You might say it's what's between you and the hardware, but that would cover pretty much all software. So let's say it's the software that sits between your software and the hardware. But does that mean that the library you picked up from some web site is part of the operating system? We probably want our operating-system definition to be a bit less inclusive. So, let's say that **it's that software that almost everything else depends upon**. This is still vague, but then the term is used in a rather nebulous manner throughout the industry.

Perhaps we can do better by describing what an operating system is actually supposed to do. From a programmer's point of view, **operating systems provide useful abstractions of the underlying hardware facilities**. Since many programs can use these facilities at once, **the operating system is also responsible for managing how these facilities are shared**.



There is a body of software, in fact, that is responsible for making it easy to run programs (even allowing you to seemingly run many at the same time), allowing programs to share memory, enabling programs to interact with devices, and other fun stuff like that. That body of software is called the **operating system**, as it is in charge of making sure the system operates correctly and efficiently in an easy-to-use manner.



Do we like any of
these definitions?

def·i·ni·tion  **noun** \,de-fə-'ni-shən\

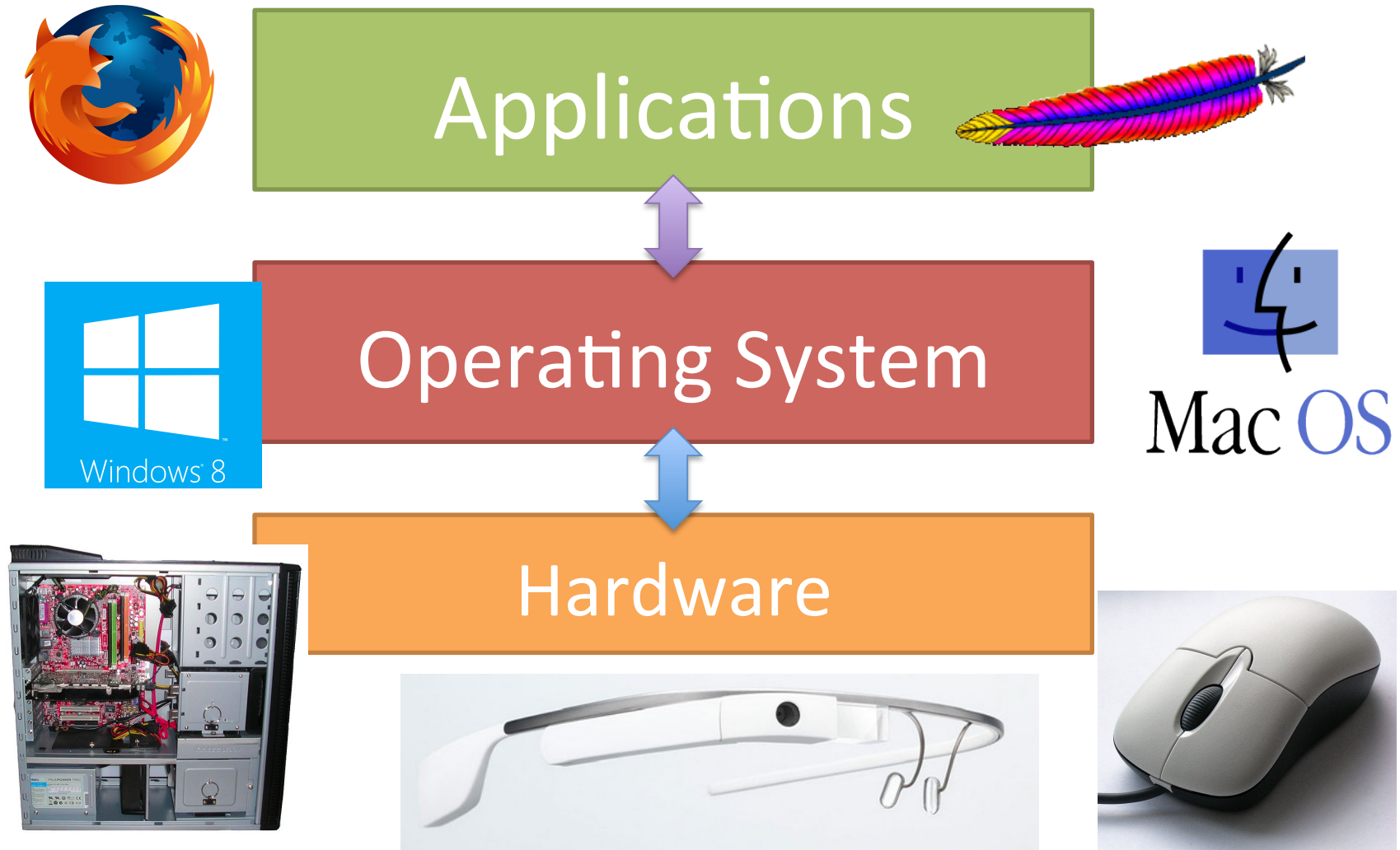
: an explanation of the meaning of a word, phrase, etc. :
a statement that defines a word, phrase, etc.

: a statement that describes what something is

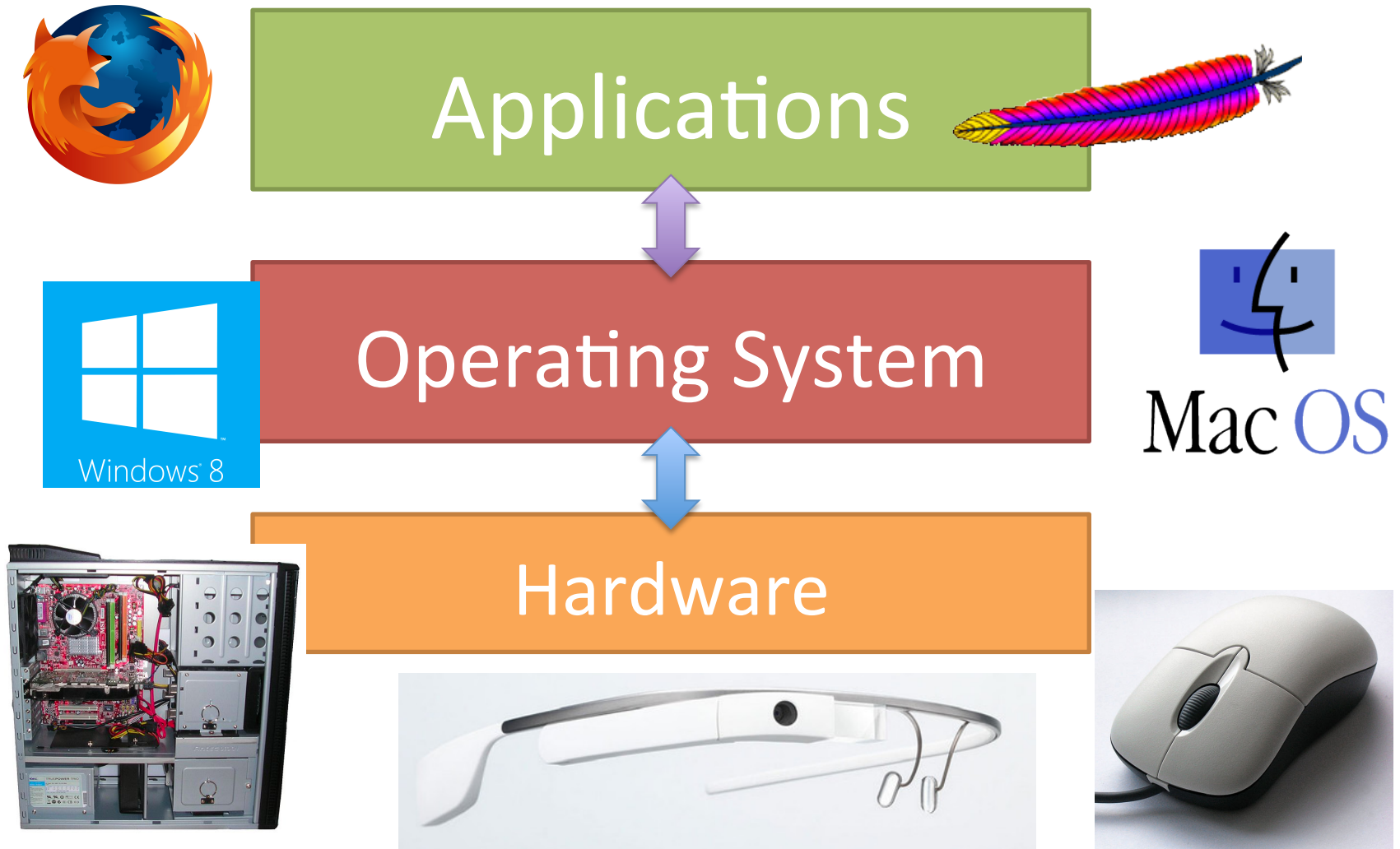
: a clear or perfect example of a person or thing

<http://www.merriam-webster.com/dictionary/definition>

Simplistic View of Computing Systems



More Realistic View



cs4414 OS Definition

An operating system is a program that manages resources and provides abstractions.

Main Ideas in cs4414

Managing Resources

How do you share processors, memory, and hardware devices among programs?

Providing Abstractions

How do you provide programs with clean and easy to use interfaces to resources, without sacrificing (too much) efficiency and flexibility?

Does it have an Operating System?



Apple II
1977





- ✗ Manage Resources
- ✓ Provide Abstractions

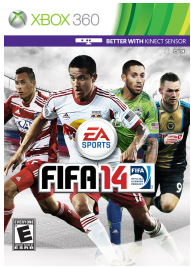




Not just one operating system, but dozens of them!

Does it have an Operating System?

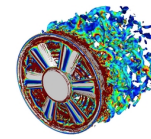
Course Overview



Google



DuckDuckGo

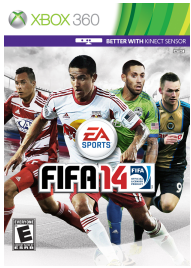


Cool Computing Stuff

Its all magic!

Physics

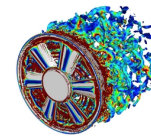
Minimizing Magic



Google



DuckDuckGo



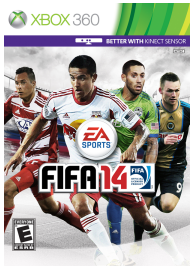
Cool Computing Stuff

Its all magic!

Four Years Studying
Computing at an
Elite Public
University

Physics

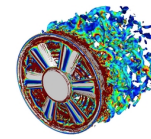
Minimizing Magic



Google



DuckDuckGo



Cool Computing Stuff

Its all magic!

Four Years Studying
Computing at an
Elite Public
University

Its all understandable!
(and I can do magical things!)

Physics

Minimizing Magic



Cool Computing Stuff

Its all magic!

cs1110

By the time you graduate,
nothing should be “magic”
other than **how transistors
work** and **NP-Completeness**.

Physics

Minimizing Magic

Cool Computing Stuff

Its all magic!

cs1110

cs2110

cs3102

cs2150

cs3330

cs2150

cs2330

By the time you graduate,
nothing should be “magic”
other than **how transistors
work** and **NP-Completeness**.

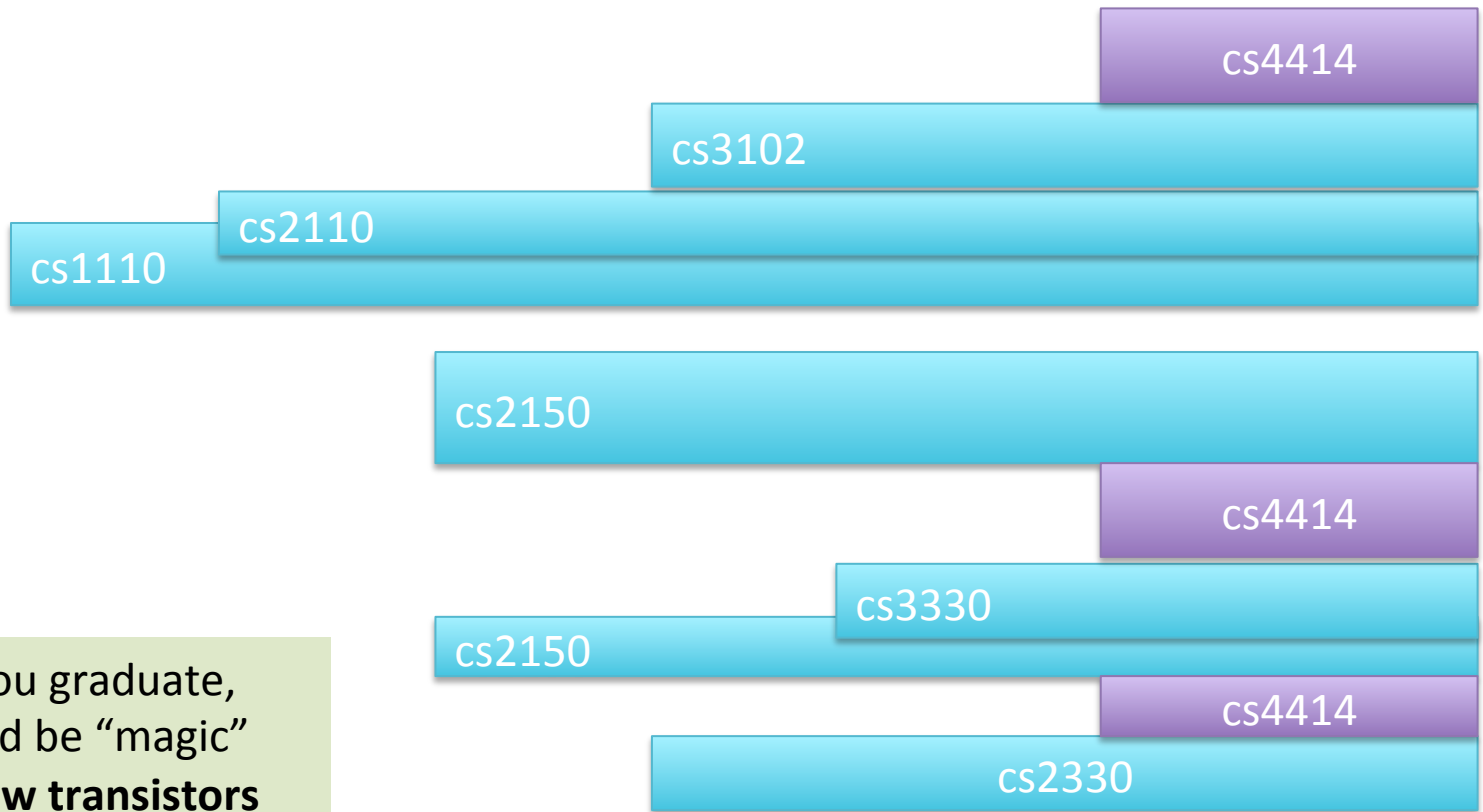
Physics

Minimizing Magic

Cool Computing Stuff

It's all magic!

By the time you graduate,
nothing should be “magic”
other than **how transistors
work** and **NP-Completeness**.



Physics

Minimizing Magic

Cool Computing Stuff

It's all magic!

By the time you graduate,
nothing should be “magic”
other than **how transistors
work** and **NP-Completeness**.

electives

cs4414

cs3102

cs2110

cs1110

cs4610

cs2150

cs4414

cs3330

cs2150

cs4414

cs2330

Physics

Minimizing Magic

(Academic) Goal of the Class

Improve our
understanding of how
computers work.

(Academic) Goal of the Class

Improve our
understanding of how
computers work.

Not:

Why bother?

Why bother?

1. Better understanding of how computers work make you a **better programmer** which will help you build something cool, succeed in grad school, or get a more interesting job.

Why bother?

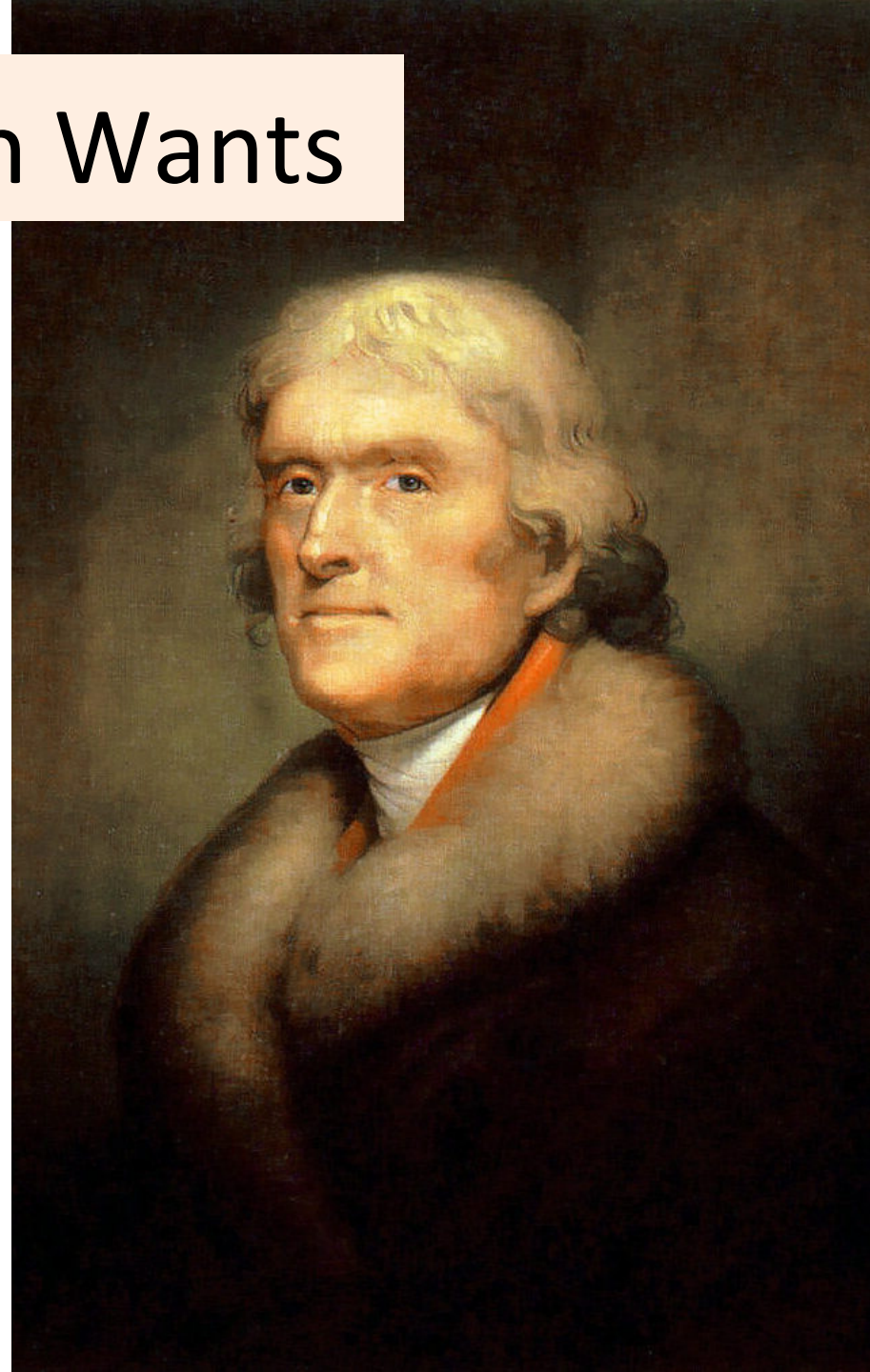
1. Better understanding of how computers work make you a **better programmer** which will help you build something cool, succeed in grad school, or get a more interesting job.
2. Better understanding of how computers work (and why) is **intellectually, culturally, and scientifically interesting.**

Why bother?

1. Better understanding of how computers work make you a **better programmer** which will help you build something cool, succeed in grad school, or get a more interesting job.
2. Better understanding of how computers work (and why) is **intellectually, culturally, and scientifically interesting.**

If these reasons don't apply for you and you are only in this class because there is a **bureaucratic requirement** that you take it so some Dean will hand you a nice bit of paper in front of your parents, you should meet with me to figure out an alternative.

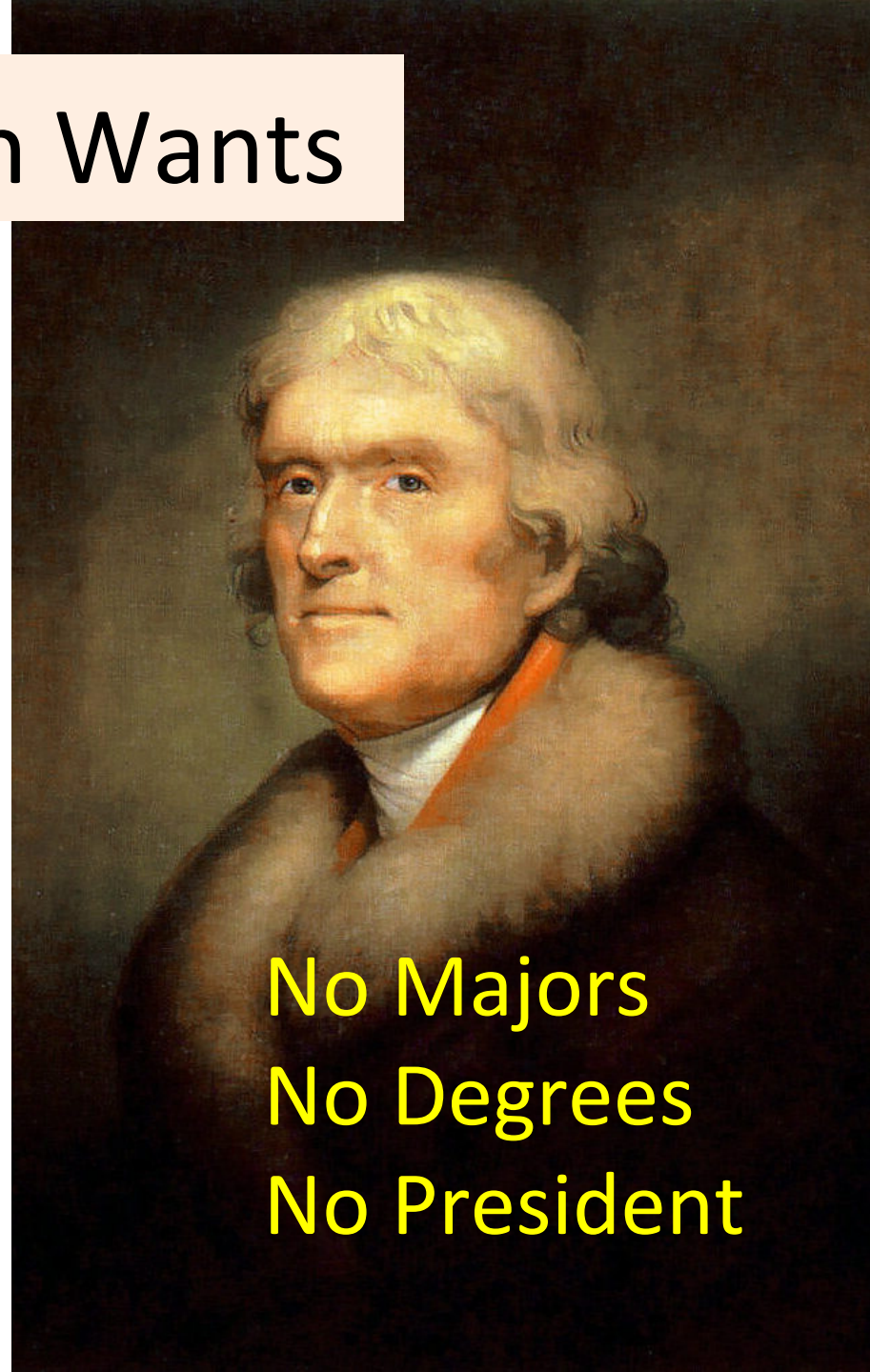
What Mr. Jefferson Wants



What Mr. Jefferson Wants

“We wish to establish in the upper country of Virginia, and more centrally for the State, a **University** on a plan **so broad and liberal and modern**, as to be **worth patronizing with the public support**, and be a **temptation to the youth of other States to come and drink...**”

TJ's letter to Joseph Priestly, 1800



No Majors
No Degrees
No President

Note: this does not mean he wants you to be lazy:

Thomas Jefferson enrolled in the College of William and Mary on March 25, 1760, at the age of 16... By the time he came to Williamsburg, the young scholar was proficient in the classics and able to read Greek and Latin authors in the original... He was instructed in natural philosophy (**physics, metaphysics, and mathematics**) and moral philosophy (**rhetoric, logic, and ethics**). A keen and diligent student, he displayed an avid curiosity in all fields and, according to family tradition, **he frequently studied fifteen hours a day.**

What is my goal for lectures?

Convey some complex technical ideas

Teach you what you need to know to do the projects

Avoid being fired

Keep most of you awake for 75 minutes

Get you to laugh at dumb jokes

What is my goal for lectures?

Lectures are a *horrible* medium for learning complex ideas.
Better to read wikipedia.

Teach you what you need to know to do
the projects

Avoid being fired

Keep most of you awake for 75 minutes

Get you to laugh at dumb jokes

What is my goal for lectures?

Lectures are a *horrible* medium for learning complex ideas.
Better to read wikipedia.

The **point of the projects is to teach** you things I want you to learn in the class (mostly by suggesting things you should learn on your own).

Avoid being fired

Keep most of you awake for 75 minutes

Get you to laugh at dumb jokes

What is my goal for lectures?

Lectures are a *horrible* medium for learning complex ideas.
Better to read wikipedia.

The **point of the projects is to teach** you things I want you to learn in the class (mostly by suggesting things you should learn on your own).

I have **tenure** already

Keep most of you awake for 75 minutes

Get you to laugh at dumb jokes

What is my goal for lectures?

Lectures are a *horrible* medium for learning complex ideas.
Better to read wikipedia.

The **point of the projects is to teach** you things I want you to learn in the class (mostly by suggesting things you should learn on your own).

I have **tenure** already

You probably should be getting **more sleep!**

Get you to laugh at dumb jokes

What is my goal for lectures?

Lectures are a *horrible* medium for learning complex ideas.
Better to read wikipedia.

The **point of the projects is to teach** you things I want you to learn in the class (mostly by suggesting things you should learn on your own).

I have **tenure** already

You probably should be getting **more sleep!**

Monty Python is **funnier** (unless you are Kevin Redmon)

My Real Goal for Lectures

Provide **context** and **meaning** for the things you have or will later **learn on your own**.

(I also have an insidious goal for lectures, that I won't tell you about until later...)

Course Assignments

The Problem Sets are Suggestions

If you have a **better idea**,
convince me, and you
should **do that instead**.



Plan for Projects

PS0 (Friday): Rust tutorial and course survey

PS1 (23 Jan): `zhttp` web server

PS2 (9 Feb): `gash` (learn about processes)

PS3 (3 March): `zhttp` web server (learn about synchronization, memory management)

PS4 (2 April): hacking a (relatively simple) kernel

This generation of students got into “UVa” by doing exactly and precisely what teacher wants. If teacher is vague about what he wants, they work a lot harder to figure out what they want and whether or not it is good. **The vaguer the directions, the more likely the opportunity for serendipity to happen. It drives them nuts!**

Harvard Professor John Stilgoe
(on "60 Minutes", 4 January 2004)

Final Project

Final Project (29 April):
(almost*) anything you want!

* must be at least two of:
fun, technically interesting,
useful, relevant

Start thinking of ideas now – if you come up with something sufficiently worthwhile, can substitute for PS4/PS3/PS2/etc.

Some Examples

<http://uvasear.ch/>

Nishant Shukla
Jasdev Singh

Top UVA Searches

[Grab the extension](#) - [Home](#)

Name	Search Count	Last Searched
Kevin Edelmann	100	2014-01-03 21:34:15
Jasdev Singh	86	2014-01-10 12:14:41
Anat Gilboa	18	2013-12-31 17:23:18
Michael Recachinas	14	2014-01-01 15:25:51
Haroun Ahmed	11	2014-01-03 11:36:22
Justin Herron	6	2013-12-08 11:52:43
Shiv Sinha	5	2014-01-09 19:46:52
Matthew Kincaid	5	2013-12-06 19:18:33
Ranjana Addanki	5	2013-12-18 15:39:02
Kelvin Green	4	2013-12-05 12:06:09



Made with love by [@jasdev](#) and [@binroot](#)

Norvigtorious

Google play

Search

Apps

My apps

Shop

Games

Editors' Choice

Norvigtorious - benchmarking

Dan Nizri · December 4, 2013

Tools

Installed

This app is compatible with all of your devices.

★★★★★ (3)

8+1 +8 Recommend this on Google

BENCHMARKS

- Find View By ID
- Set Content View
- Read 1 MB From RAM
- Write 1 MB To RAM
- Read 1 MB From Internal Storage
- Write 1 MB To Internal Storage
- Read 1 MB From SD Card

View Statistics

Send Packet to Europe and Back

191,000,000 ns!

Recalculate

Encrypt 1 MB of Data

258,758,545 ns!

Recalculate

Write 1 MB To SD Card

62,072,754 ns!

Recalculate

Average Benchmark Times for Android

Benchmark Type	Average Time (ns)
Find View By ID	38,000
Read 1 MB From RAM	3,094,000
Read 1 MB From Internal Storage	3,436,000
Write 1 MB To RAM	8,528,000
Set Content View	9,281,000
Write 1 MB To Internal Storage	11,334,000
Write 1 MB To SD Card	75,213,000
Read 1 MB From SD Card	91,935,000
Sort 10,000 Strings Alphabetically	107,925,000
Send Packet to Europe and Back	194,833,000
Decrypt 1 MB of Data	285,148,000

View My Device's Statistics

Alex Fabian
Daniel Nizri
Renee Seaman
Casey Silver

IronKernel



Kevin Broderick
Alex Lamana
Zeming Lin
John Stevans
Wil Thomason

We will use this for PS4!

1. ▲ **Rust 0.9 released** (mozilla.org)
232 points by kibwen 6 hours ago | 77 comments
2. ▲ **Lost city in Darfur** (openstreetmap.org)
149 points by Noelkd 5 hours ago | 46 comments
3. ▲ **Oculus VR's New "Crystal Cove" Prototype Is Kind Of Amazing** (techcrunch.com)
46 points by kirtijthorat 2 hours ago | 31 comments
4. ▲ **Overstock.com Is Now Accepting Bitcoins** (wired.com)
246 points by ironchief 8 hours ago | 185 comments
5. ▲ **Inject JavaScript to explore native apps on Windows, Mac, Linux and iOS** (frida.re)
70 points by oleavr 4 hours ago | 33 comments
6. ▲ **Words are Hard** (hackerschool.com)
22 points by mgeraci 1 hour ago | 26 comments
7. ▲ **Frog Design's Tech Trends 2014** (frogdesign.com)
14 points by cpeterso 1 hour ago | 3 comments
8. ▲ **Tesla Model X Spotted on Public Road in Culver City, California** (insideevs.com)
33 points by kirtijthorat 2 hours ago | 17 comments
9. ▲ **Sources: We were pressured to weaken mobile security in the 80's** (aftenposten.no)
45 points by kmskontorp 4 hours ago | 7 comments
10. ▲ **Implementing a JIT Compiled Language with Haskell and LLVM** (stephendiehl.com)
156 points by rwosync 9 hours ago | 13 comments
11. ▲ **Preview the New Deployments API** (github.com)
53 points by joeyespo 5 hours ago | 8 comments
12. ▲ **Luck: The Secret Sauce of Successful Startups** (ramlijohn.com)
18 points by ramlijohn 2 hours ago | 18 comments
13. ▲ **How Silicon Valley became "The Man"** (hbr.org)
66 points by jenningsjason 6 hours ago | 21 comments

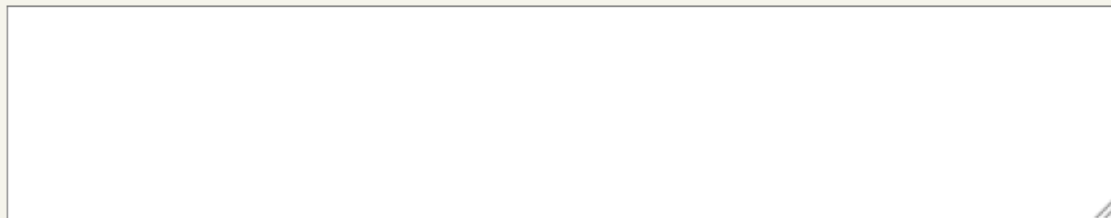
1. ▲ Rust 0.9 released (mozilla.org)

232 points by kibwen 6 hours ago | 77 comments

2. ▲ Lost city in Darfur (smarterhacker.com)

3. ▲ Rust 0.9 released (mozilla.org)

232 points by kibwen 6 hours ago | 77 comments



add comment

9 ▲ kibwen 5 hours ago | link

A selection of some of my favorite aspects of this release:

1. The (yet-ongoing) removal of managed pointers, leaving us with one fewer pointer type. Final tally of built-in pointer types: unique pointers, mutable references, and immutable references.
2. The dead code detection pass (<https://github.com/mozilla/rust/pull/10477>), contributed by a student of the University of Virginia's Rust-based systems programming class (<http://rust-class.org/pages/using-rust-for-an-undergraduate-...>).
3. The `Any` trait, giving us on-demand dynamic typing (<https://github.com/mozilla/rust/pull/9967>).
4. The clean abstraction of green threads and native threads out into their own libraries (<https://mail.mozilla.org/pipermail/rust-dev/2013-December/00...>) such that any library that makes use of the stdlib will work regardless of which strategy the user selects.

1. ▲ Rust 0.9 released (mozilla.org)
232 points by kibwen 6 hours ago | 77 comments

mozilla / rust

Watch 355 Star 3,869 Fork 795

Closed

ktt3ja wants to merge 2 commits into mozilla:master from ktt3ja:dead-code

1,905 #10477

Conversation

Commits 2

Files Changed 100



ktt3ja opened this pull request 2 months ago

Add dead-code warning pass

No one is assigned

No milestone

PR for issue #1749 mainly to get some feedback and suggestion. This adds a pass that warns if a function, struct, enum, or static item is never used. For the following code,

```
pub static pub_static: int = 0;
static priv_static: int = 0;
static used_static: int = 0;

pub fn pub_fn() { used_fn(); }
fn priv_fn() { let unused_struct = PrivStruct; }
fn used_fn() {}

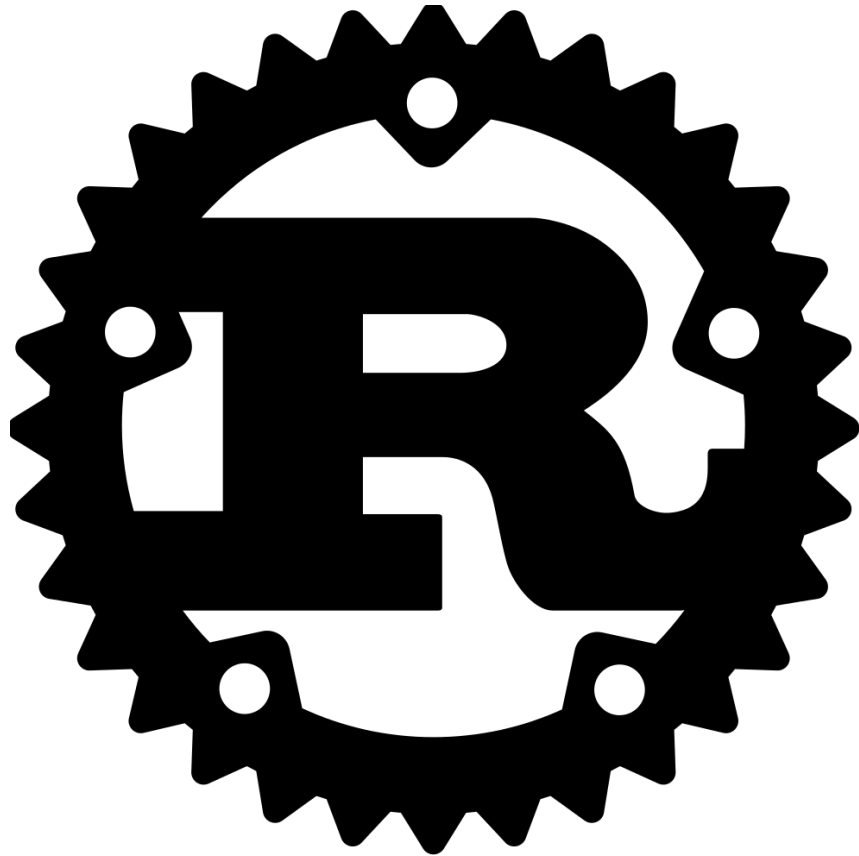
pub struct PubStruct();
struct PrivStruct();
struct UsedStruct1 { x: int }
struct UsedStruct2(int);
struct UsedStruct3();
```

Closed

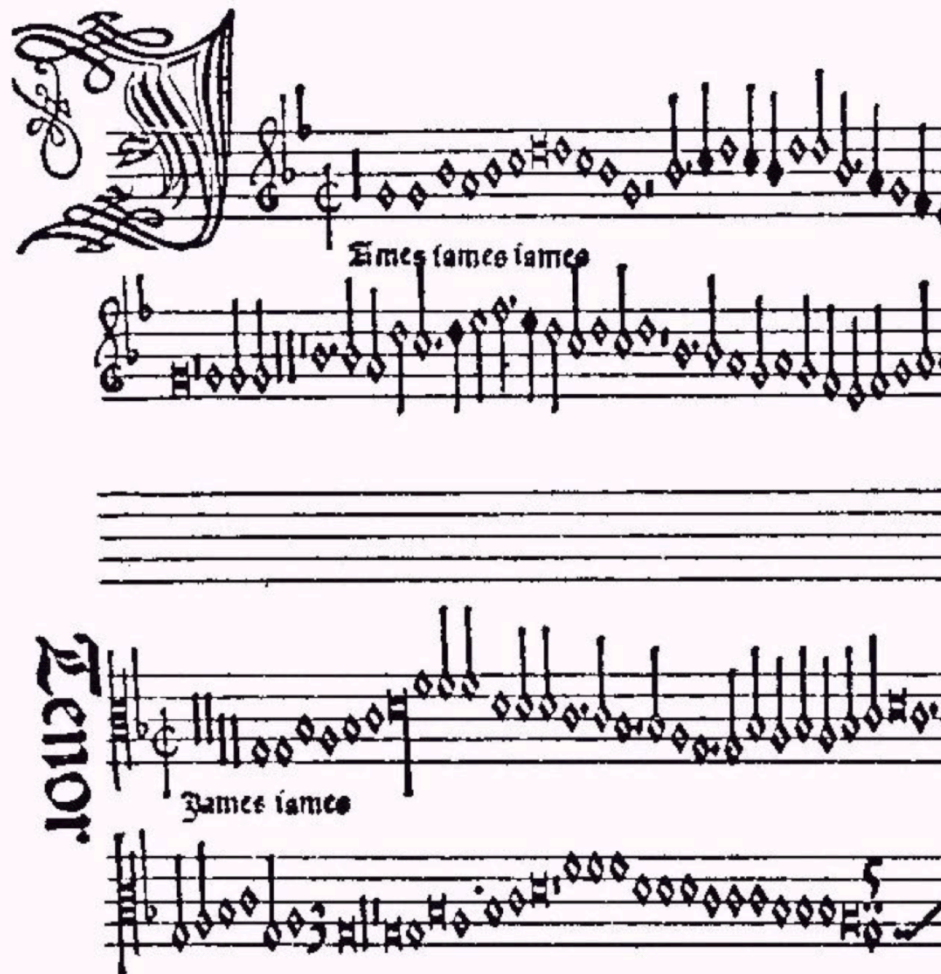
+ 722 additions

- 1,183 deletions

Kiet Tran



Why learn a
new
programming
language?

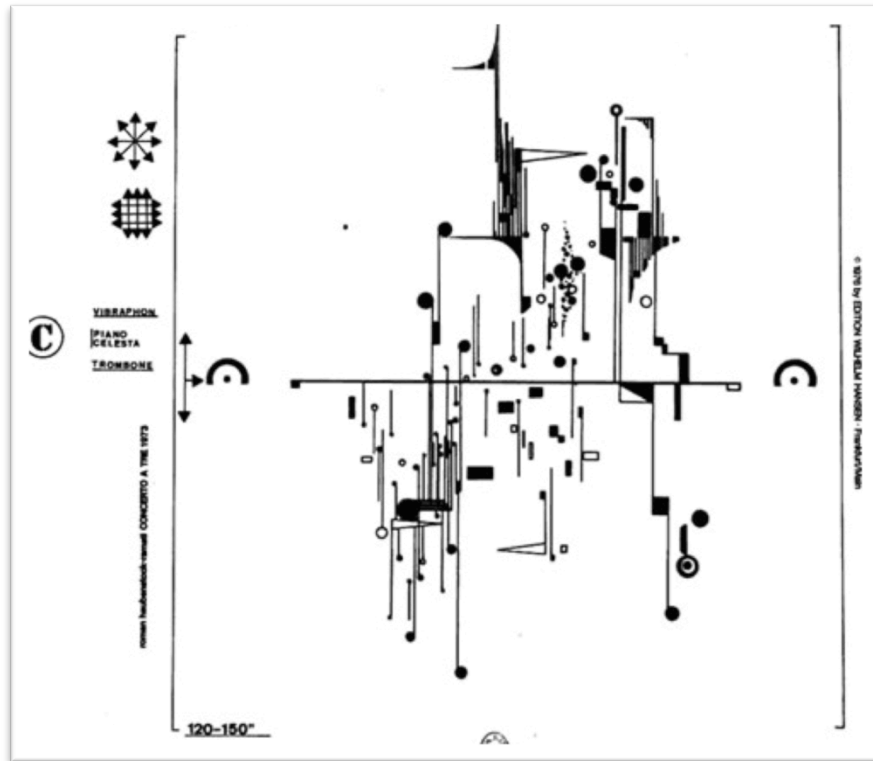


“Jamais Jamais Jamais” from
Harmonice Musices Odhecaton
A. (1501)

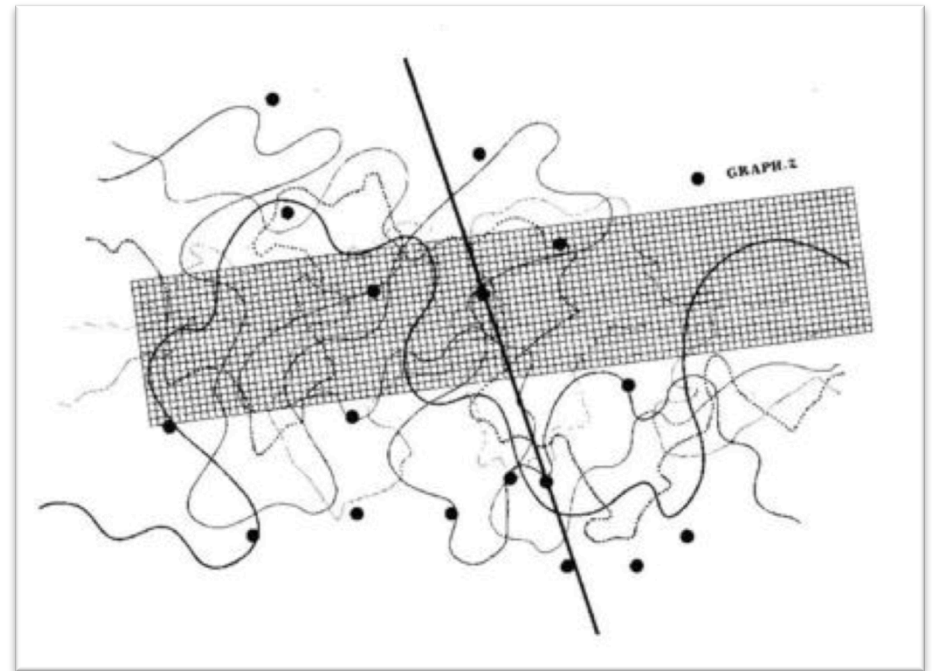
J S Bach, “Coffee Cantata”,
BWV 211 (1732)

[www.npj.com/homepage/teritowe/
jsbhand.html](http://www.npj.com/homepage/teritowe/jsbhand.html)

Modern Music Notation



Roman Haubenstock-Ramati, *Concerto a Tre*



John Cage, *Fontana Mix*

<http://www.medienkunstnetz.de/works/fontana-mix/audio/1/>

I

TACET

II

TACET

III

TACET

I

TACET

II

TACET

III

TACET

4'33"

FOR ANY INSTRUMENT OR COMBINATION OF INSTRUMENTS

John Cage

COPYRIGHT © 1960 BY KENNETH PESS INC., 373 PARK AVENUE, N.Y., N.Y. 10016

NOTE: THE TITLE OF THIS WORK IS THE TOTAL LENGTH IN MINUTES AND SECONDS OF ITS PERFORMANCE. AT WOODSTOCK, N.Y., AUGUST 29, 1952, THE TITLE WAS 4'33" AND THE THREE PARTS WERE 33', 2'40", AND 1'20". IT WAS PERFORMED BY DAVID TUDOR, PIANIST, WHO INDICATED THE BEGINNINGS OF PARTS BY CLOSING, THE ENDINGS BY OPENING, THE KEYBOARD LID. AFTER THE WOODSTOCK PERFORMANCE, A COPY IN PROPORTIONAL NOTATION WAS MADE FOR IRWIN KREMER. IN IT THE TIMELONGTHS OF THE MOVEMENTS WERE 30", 2'23", AND 1'40". HOWEVER, THE WORK MAY BE PERFORMED BY ANY DISTINGUISHED TALENT(S) AND THE MOVEMENTS MAY LAST ANY LENGTHS OF TIME.

Thought and Action

Languages change the way we **think**

BASIC: think about GOTO

Algol, Python: think about assignments, control blocks

Scheme, Haskell: think about procedures

Java, C++: think about types, objects

Languages provide **abstractions** of machine resources

- Hide dangerous/confusing details: memory locations, instruction opcodes, number representations, calling conventions, etc.
- Hiding more increases **simplicity**, but limits **control**

Why so many programming languages?



The Pragmatic Programmers

Programming
Ruby
The Pragmatic Programmers' Guide



Dave Thomas
with Chad Fowler and Andy Hunt



SECOND EDITION
THE
C
ANSI C
PROGRAMMING
LANGUAGE

BRIAN W. KERNIGHAN
DENNIS RITCHIE

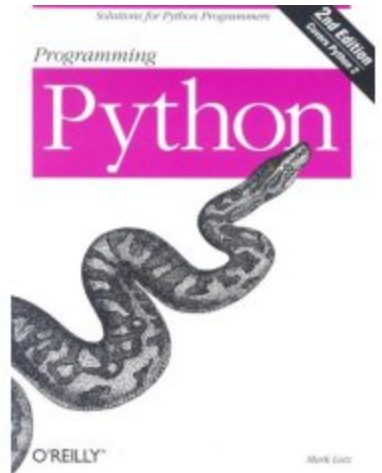


THE DEFINITION
OF STANDARD ML
(REVISED)

Robert Milner
Mads Tarsni
Simon Thompson



BJARNE
STROUSTRUP
The Creator of C++



Fundamental Differences

Fundamental Differences

All equivalently powerful!

- *Universal languages*: all capable of simulating each other

Fundamental Differences

All equivalently powerful!

- *Universal languages*: all capable of simulating each other

Fundamental differences

Expressiveness: how easy it is to describe a computation

Control: how much programmer can control machine

“Truthiness”: likelihood program means programmer wants

Safeness: minimize impact of programmer mistakes

Difficult to achieve all of these at once

Fundamental Differences

All equivalently powerful!

- *Universal languages*: all capable of simulating each other

Fundamental differences

Expressiveness: how easy it is to describe a computation

Control: how much programmer can control machine

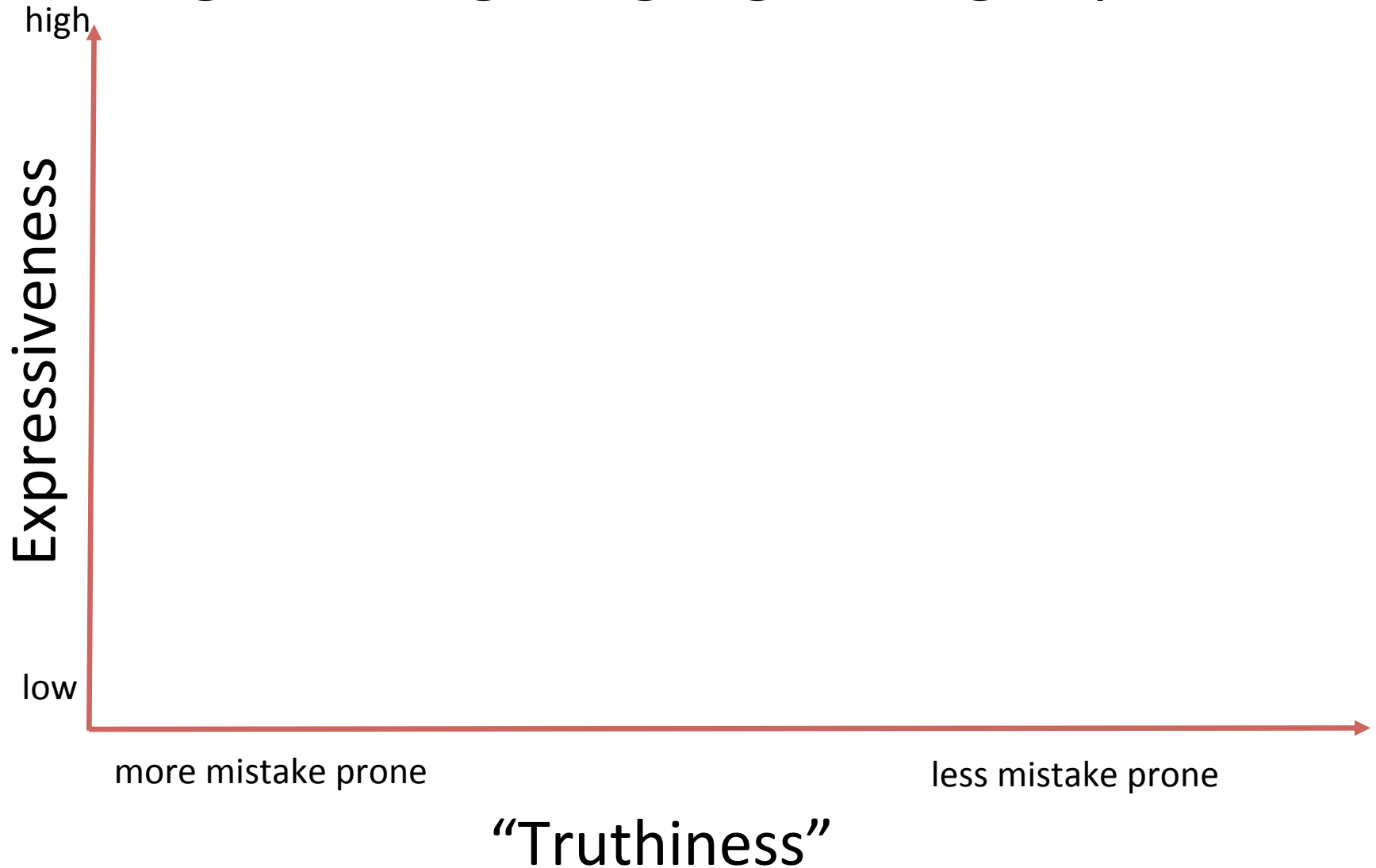
“Truthiness”: likelihood program means programmer wants

Safeness: minimize impact of programmer mistakes

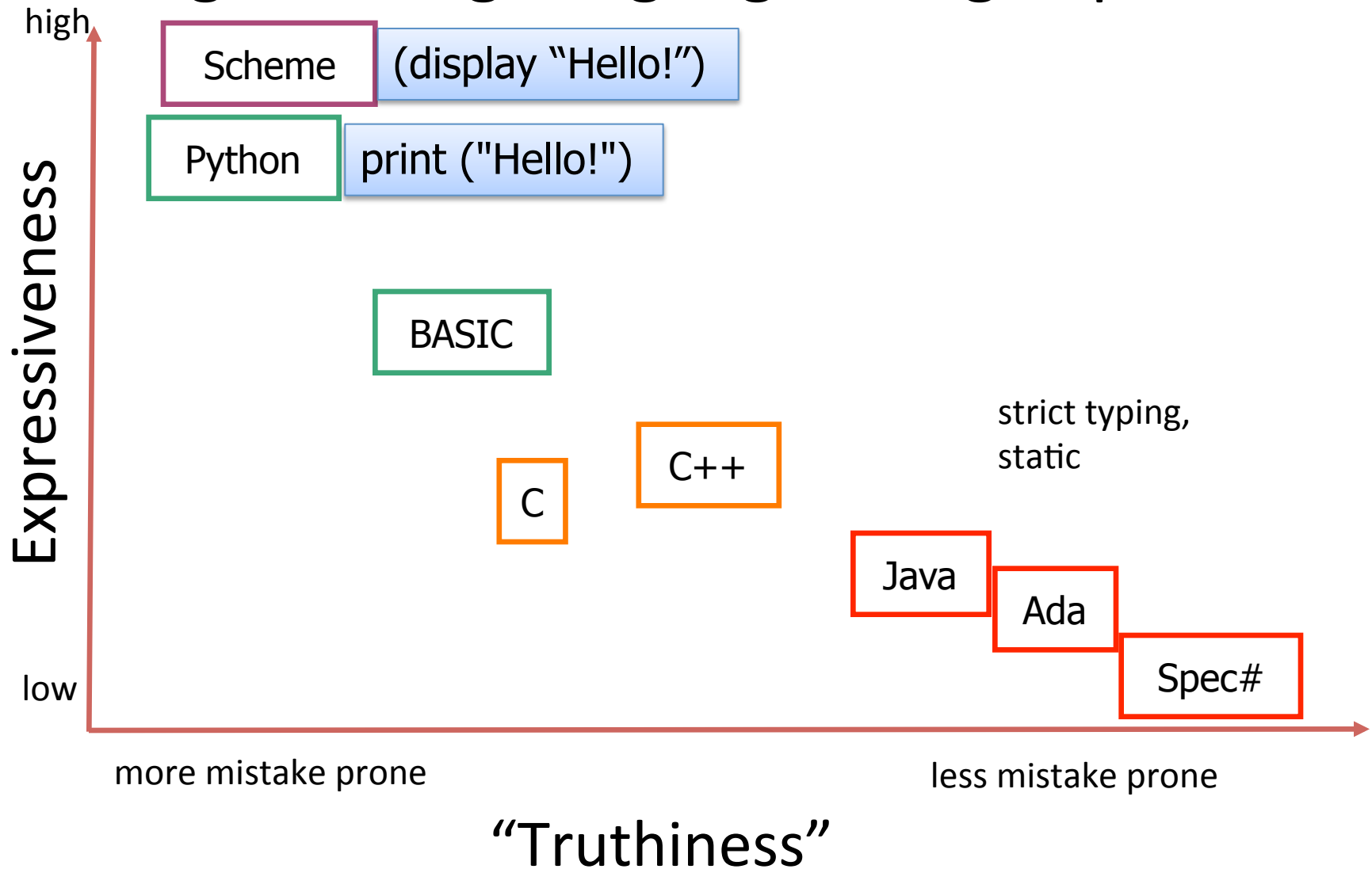
Difficult to achieve all of these at once

What do we want for systems programming?

Programming Language Design Space



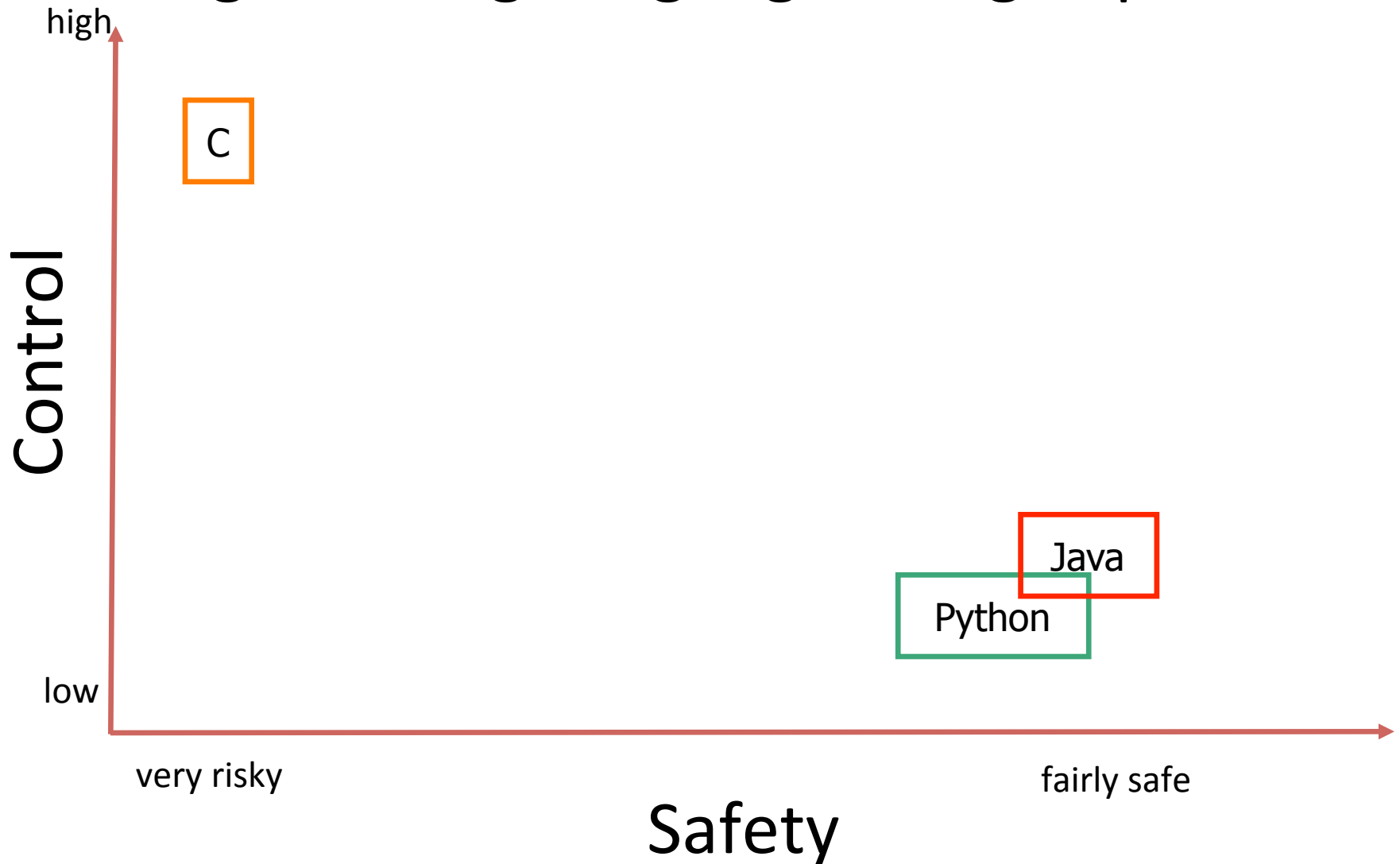
Programming Language Design Space



Programming Language Design Space



Programming Language Design Space



“a safe, concurrent, practical language”

Its design is oriented toward concerns of “programming in the large”, that is, of creating and maintaining boundaries – both abstract and operational – that preserve large-system integrity, availability and concurrency.

from <http://www.rust-lang.org/>



Rust

Advances in programming language design and compiler implementation make it possible to get both control and safety, and mostly get expressiveness and “truthiness” all at the same time!

Charge

Bring a laptop to use in Thursday's class!

- Do (or at least attempt) everything on the Class 1 notes **Action Items** before class Thursday
 - Download and setup your computing environment for cs4414
 - Setup your github account
- Next class:
 - Help getting everything working on your machine
 - Help getting started with Rust (Rust tutorial)