CSCI 334:
Principles of Programming Languages

Lecture 21: Object-oriented programming

Instructor: Dan Barowy

Williams

Topics

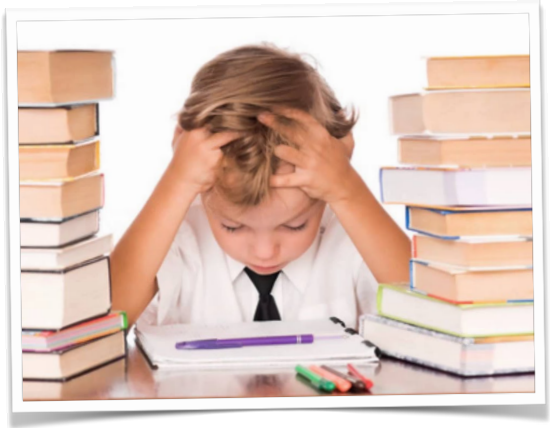
Object-oriented programming

Your to-dos

1. Last quiz, **due tomorrow, 12/7**.
2. Please bring laptop/tablet **next class** to fill out student course surveys.
3. Project checkpoint #3, "mostly working," **due Sunday 12/11**.
4. Final project due **Sunday 12/18**.

Object-Oriented Programming

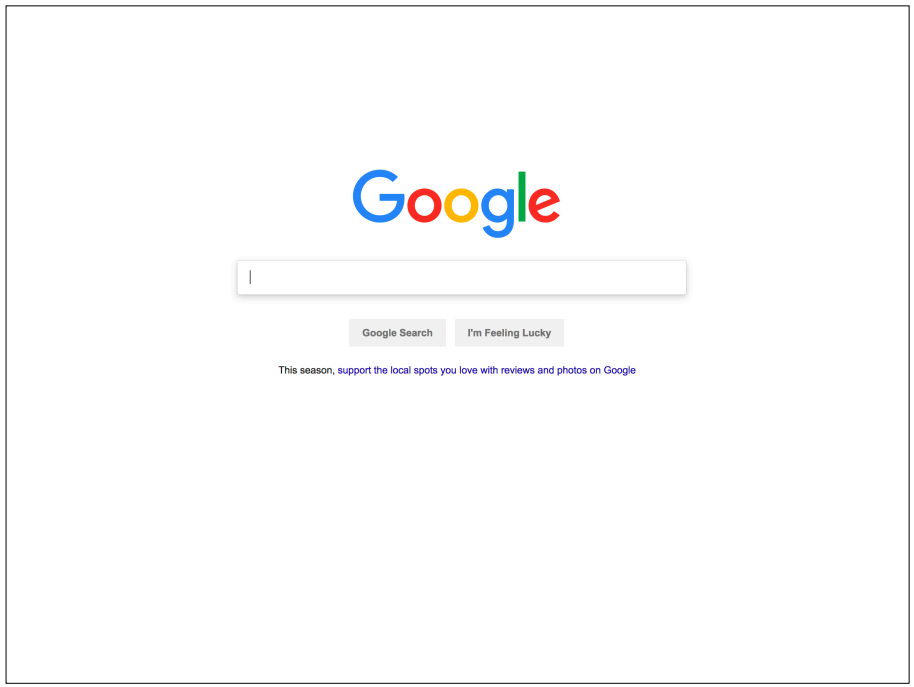
Programming in the small



Programming in the large


Ruby on Rails
Sustainable productivity for web-application development

TensorFlow



What languages?

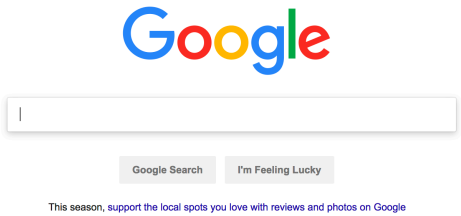
Java

C++

Ruby

TensorFlow

C++, Python



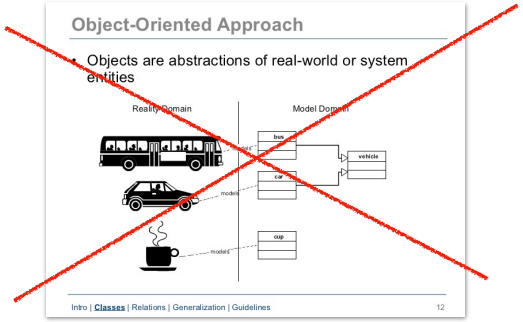
Java

Object-Oriented Programming

- OOP is both a language design philosophy and a way of working (OO design).
- OOP is possibly the most impactful development in the history of programming languages.

What OOP is Not

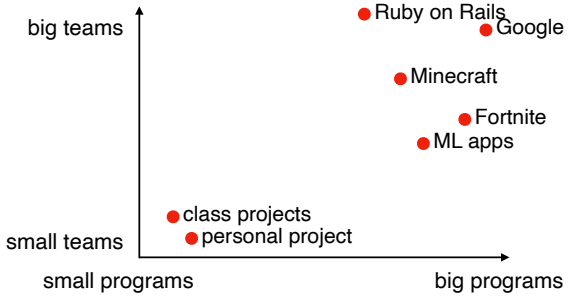
- Many, many instructors introduce OOP as a way of naturally simulating the world.



- This misses the point of OOP entirely!

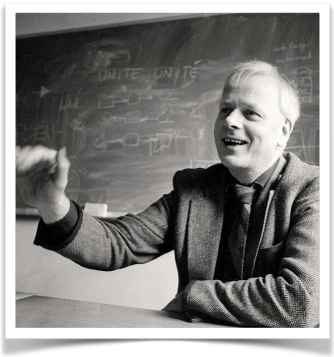
What OOP is

- Object-oriented programming is actually about scalability.
- Scalability in codebase size was the original motivation.
- But OO philosophy also has had a big effect on the scalability of programming teams.



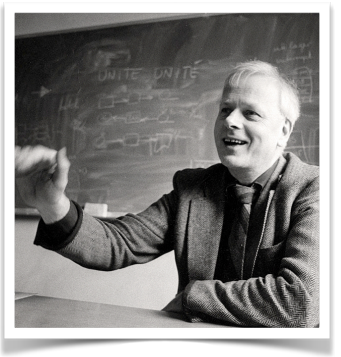
History

- First language recognizable as OO: Simula-67.
- Developed by Kristen Nygaard and others at the Norwegian Computing Center.
- Grew out of frustrations using ALGOL.
- Original plan was to add an “object” library, inspired by C.A.R. Hoare’s “record classes”.
- It was eventually realized that objects were a fundamentally different way of structuring a program; Simula became its own language.

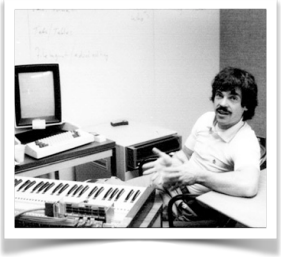


History

- But Simula-67 was not the most influential OO language.
- That language was…



Smalltalk



Alan Kay
Essentially invented the laptop/tablet ("Dynabook")

Turing Award

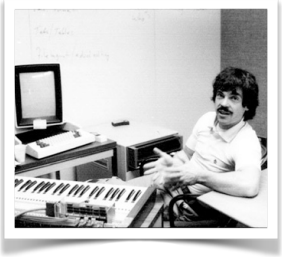
Dan Ingalls
Essentially invented object oriented programming

Grace Murray Hopper Award

Adele Goldberg
Essentially invented graphical user interfaces

ACM Software Systems Award

Smalltalk

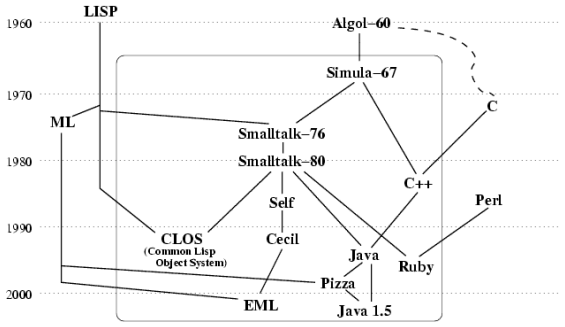


- First mainstream OO success: Smalltalk
- Developed by Alan Kay, Dan Ingalls, and Adele Goldberg at Xerox PARC and later Apple Computer.
- Used to implement major components of the groundbreaking Xerox Alto computer: OS, compiler, GUI, applications.
- Highly influential. E.g., C++, Java, Ruby, etc.

Smalltalk

And they showed me really three things. But I was so blinded by the first one I didn't even really see the other two. One of the things they showed me was object orienting programming they showed me that but I didn't even see that. The other one they showed me was a networked computer system… they had over a hundred Alto computers all networked using email etc., etc. I didn't even see that. I was so blinded by the first thing they showed me which was the graphical user interface… within you know ten minutes it was obvious to me that all computers would work like this some day.

Smalltalk



OK, really, what is OO?

Object-oriented programming is composed primarily of four key language features:

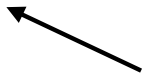
1. Abstraction
2. Dynamic dispatch
3. Subtyping
4. Inheritance

Purpose: polymorphism at scale

OK, really, what is OO?

Object-oriented programming is composed primarily of four key language features:

1. Abstraction
2. **Dynamic dispatch**
3. Subtyping
4. Inheritance



In my mind, this is OO's killer feature.

Purpose: polymorphism at scale

Object-oriented programming is a solution to complexity

(video)

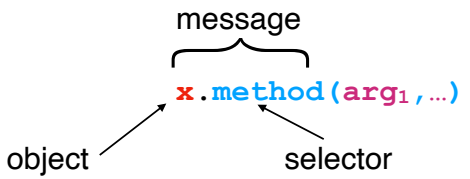
(code)

Dynamic Dispatch

(the secret to understanding how Java, Python, Ruby, etc. work)

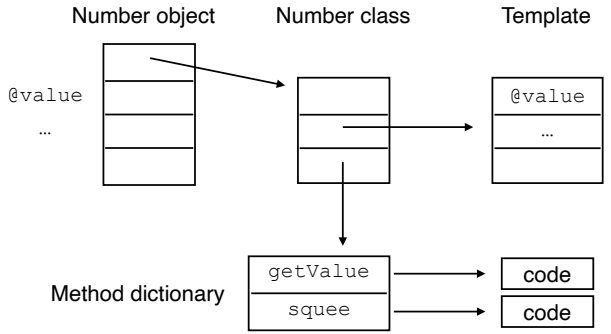
Dynamic Dispatch

- **Dynamic dispatch** is the OO mechanism for **polymorphism**.
- Functions ("**methods**") are always bound to an object (or class)
- A method is called ("**dispatched**") by sending a "**message**" to the "**selector**" of an object.



Dynamic Dispatch

- Dynamic dispatch is an **algorithm** for finding a the **implementation** for a given **selector** (i.e., method).



Recap & Next Class

This lecture:

OOP

Next lecture:

Student Course Surveys

How to give a good technical talk