CSE341: Programming Languages

Lecture 21
Late Binding;
OOP as a Racket Pattern

Dan Grossman

Fall 2011

Today

Dynamic dispatch aka late binding aka virtual method calls

- Call to **`self.m2()`** in method **`m1`** defined in class **`C`** can *resolve to* a method **`m2`** defined in a subclass of **`C`**
- Most unique characteristic of OOP

Need to define the semantics of objects and method lookup as carefully as we defined variable lookup for functional programming

Then consider advantages, disadvantages of dynamic dispatch

Then encoding OOP / dynamic dispatch with pairs and functions

- In Racket
- Complement Lecture 9's encoding of closures in Java or C

Resolving identifiers

The rules for "looking up" various symbols in a PL is a key part of the language's definition

- So discuss in general before considering dynamic dispatch

- ML: Look up variables in the appropriate environment
  - Key point of closures' lexical scope is defining "appropriate"
  - Field names (for records) are different

- Racket: Like ML plus let, letrec, and hygienic macros

- Ruby:
  - Local variables and blocks mostly like ML and Racket
  - But also have instance variables, class variables, and methods (all more like record fields)