CSE 341: Programming Languages

Hal Perkins

Spring 2011

Lecture 22— Multiple Inheritance, Interfaces, Mixins

Today

Have seen OO's essence: inheritance, overriding, dynamic-dispatch.

What if we want these things from more than "exactly 1 superclass"?

- *Multiple inheritance*: allow > 1 superclasses
  - Useful but has some problems (see C++)
- Java-style *interfaces*: allow > 1 *types*
  - "Irrelevant" in a dynamically typed language, but fewer problems
- *Mixins*: allow > 1 "sources of methods"
  - Close to multiple inheritance; almost as useful with fewer (?) problems
  - In Ruby

Multiple Inheritance

If code reuse via inheritance is so useful, why not allow multiple superclasses?

- Because it causes some semantic awkwardness and implementation awkwardness (we'll discuss only the former)
- (With static typing, there are some more issues)

Is it useful? Sure: A simple example is "3DColorPoint" assuming we already have "3DPoint" and "ColorPoint".

Naive view: Subclass has all fields and methods of all superclasses