327: Object-oriented programming

Lecture 6

9/20/2021

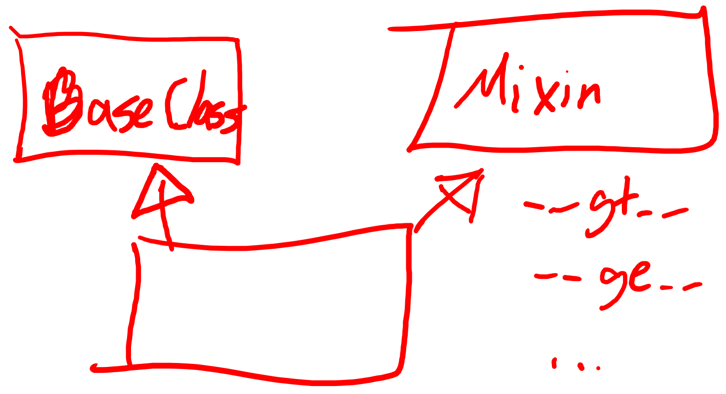
Professor Barron

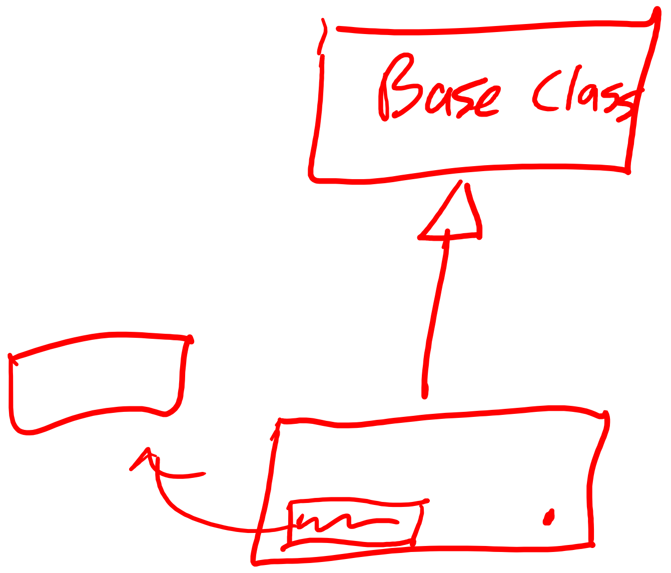
Today…

- Returning to examples of mixins and multiple inheritance
- Abstract classes and interfaces
- Python/Java/C++ comparisons
- Moving on to chapter 4…
- Exceptions

Mixins

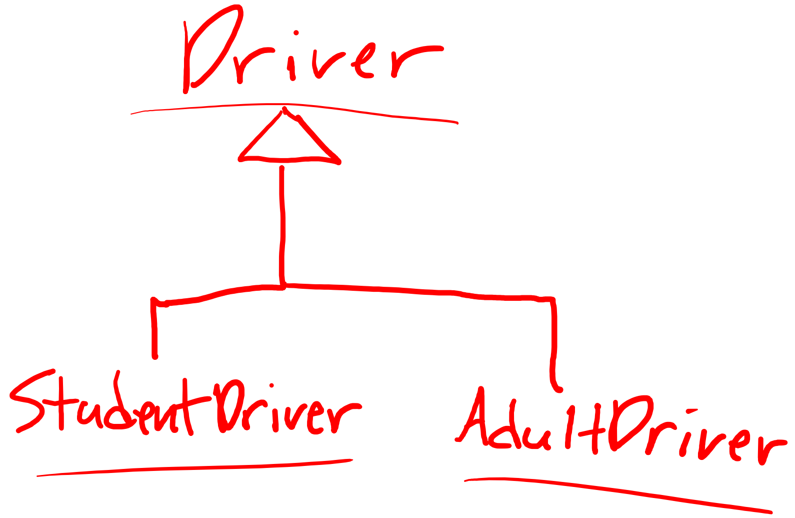
- Specific use case for multiple inheritance
- Not meant to be used on their own
- Add features to multiple other classes
  - Careful about collisions
- Composition is an alternative





Abstract classes and interfaces

- Class that is not meant to be instantiated directly
- Template for other classes
- Defines the public interface
- Typically does not implement methods
- Require subclasses to implement them
- Important in statically typed languages where polymorphism is necessary to work on different types that implement the same interface
- Not a core feature of Python (see example)
  - Still useful to structure/organize code and make it harder to forget to implement part of the interface



Other languages: Java

- Has abstract classes and interfaces
  - Very similar tools, but interfaces are a bit more restricted
  - No multiple inheritance
  - Can implement more than one interface (not usually an issue)
  - Interface with default methods…
    - have to explicitly call, like Python without using super()
    - InterfaceName.super.methodCall()
  - has super like Python, but cannot “skip a level”

Other languages: C++

- More similar to Python
- No explicit abstract classes or interfaces
    - create a “pure virtual function”
    - compile error if used without overriding first
- Supports multiple inheritance
    - compile error if ambiguous
- No super
    - diamond problem solved with “virtual inheritance”
    - can choose correct parent method with :: operator