LECTURE 6

OBJECT-ORIENTED PROGRAMMING

SUBCLASSES AND INHERITANCE II

MCS 275 Spring 2022

David Dumas

LECTURE 6: SUBCLASSES AND INHERITANCE II

Course bulletins:

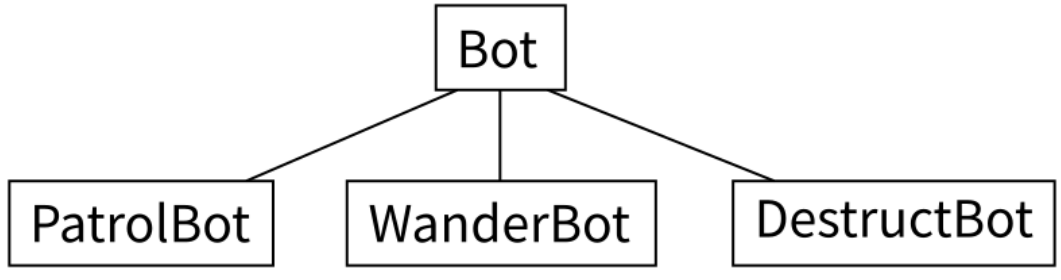
- Homework 2 is due at Noon tomorrow (Tue Jan 26).
- Project 1 will be posted today, due Fri 4 Feb at 6pm.
- Wear a well-fitting mask completely covering nose and mouth when in classroom.

PLAN

Finish our robot simulation class hierarchy

Discuss more OOP theory & practice

PLANNED BOT HIERARCHY



- `PatrolBot` walks back and forth.
- `WanderBot` walks about randomly.
- `DestructBot` sits in one place for a while and then self-destructs.

CLASS ATTRIBUTES

Attributes declared in the class definition, outside of any method, are **class attributes**.

Class attributes are shared by every instance of the class. Often used for constants.

Contrast with the **instance attributes** we have used thus far (e.g. `self.x = 1` in constructor) which exist separately for each instance.

FOUR PILLARS OF OOP

- **Encapsulation** - Classes manage their own private, internal state.
- **Abstraction** - Method calls express intent (independent of implementation).
- **Inheritance** - Distinct classes can share behavior.
- **Polymorphism** - Code using a class will also work on its subclasses.

EXTENDING THE SIMULATION

Beyond adding more robot types, how might me improve or extend the simulation?

EXTENDING THE SIMULATION

Might create a class `Arena` that manages the list of bots and the space in which they move. Would have a single `.update()` method that updates all bots.

Each bot would know what `Arena` it is in, and could call methods of `Arena` to interrogate surroundings (e.g. avoid collision, seek other bots, ...)

REFERENCES

- I discussed inheritance in MCS 260 Fall 2021 Lecture 27
- See Lutz, Chapter 31 for more discussion of inheritance.
- Lutz, Chapters 26-32 discuss object-oriented programming.

REVISION HISTORY

- 2022-01-24 Initial publication